A Formal Theory of Indexical Knowledge and Action

Yves Lespérance

Technical Report CSRI-248
February 1991

Computer Systems Research Institute
University of Toronto
Toronto, Canada
M5S 1A1

A Formal Theory of Indexical Knowledge and Action

Yves Lespérance

Department of Computer Science

University of Toronto

Toronto, Ontario, Canada

February 1991

A Thesis submitted in conformity with the requirements

for the degree of Doctor of Philosophy in the

University of Toronto

Abstract

Agents act upon and perceive the world from a particular perspective. It is important to recognize this relativity to perspective if one is not to be overly demanding in specifying what they need to know in order to be able to achieve goals through action. An agent may not know where he is, what time it is, which objects are around him, what the absolute positions of these objects are, and even who he is, and still be able to achieve his goals. This is because the knowledge required is indexical knowledge, knowledge about how one is related to things in one's environment or to events in one's history.

This thesis develops a formal theory of knowledge and action that handles the distinction between indexical and objective knowledge and allows a proper specification of the knowledge prerequisites and effects of action. The theory is embodied in a logic. The semantics of knowledge proposed is a natural extension of the standard possible-world semantic scheme. The notion of "ability to achieve a goal by doing an action" is formalized within the logic; the formalization does not require an agent to know in an absolute sense who he is or what time it is.

We then formalize various domains within the logical system proposed. These examples show how actions can be specified so as to avoid making excessive requirement upon the knowledge of agents, and how such specifications can be used to prove that an agent is able to achieve a goal by doing an action if he knows certain facts. We direct a significant part of our formalization efforts at a robotics domain, since this kind of application provides the most intuitive examples of situations where indexical knowledge is sufficient for ability. But we also formalize other domains involving temporal knowledge, knowledge of the phone system, and knowledge of one's position in a data structure. The examples involved show that the notion of indexical knowledge is more abstract than one might first imagine. On the basis of evidence provided by the temporal examples, we provide an argument to the effect that the distinction between indexical and objective knowledge accommodated by our framework is of practical interest and that it cannot be handled within previous theories.

Acknowledgements

First of all, I want to thank Hector Levesque, who in his role as co-supervisor of this thesis, infected me with his enthusiasm, provided a steady supply of encouragement, as well as invaluable technical advice. Without his efforts, this thesis would probably never have been completed.

I also want to thank Graeme Hirst, my other co-supervisor, for his continuing encouragement during the long gestation period of this thesis and his careful reading of numerous drafts. I would also like to express my gratitude to my external examiner, David Israel, for his unusually detailed and carefully considered suggestions and criticisms of the thesis. I cannot think of how someone could do a better job of appraising a thesis. Many thanks to the other members of my committee, professors Mylopoulos, Jepson, Borodin, and Urquhart, for their useful suggestions and stimulating questioning.

Numerous fellow graduate students have contributed to this work by their criticisms, discussions, and encouragement; I will mention Craig Bouthiler, Jim Des Rivieres, Diane Horton, Gerhard Lakemeyer, Jose Nunes, Calvin Ostrum, Javier Pinto, and Bart Selman, but this list could go on and on. Financial support was gratefully received from the Department of Computer Science. Finally, I want to thank my family and friends for their love and support, without which I could not have persevered in this endeavor.

Contents

Chapter 1

Introduction

1.1 Motivation

Agents need knowledge to be able to achieve goals by doing actions. For example, to be able to withdraw funds at a teller machine, one must know what procedure to use (i.e., how to insert one's bank card, what keys to press, etc.), one must know what one's identification code is, one must know that the machine is in operation at the time, that one has sufficient funds, etc. It may be physically possible for someone to withdraw money with his bank card, but if he does not remember his identification code, he will not be *able* to do it. But he may be able to plan his way around his ignorance, for example, he could decide to first look on the back cover of his address book, where he has written the code; after doing this action, he will have the requisite knowledge. Agents must reason about:

- the effects various actions would have upon their knowledge

- the knowledge prerequisites of the actions they might do — the conditions under which they are *able* to achieve goals by doing actions.

A satisfactory theory of the relationship between knowledge and action would need to formalize enough aspects of these notions and how they relate to support reasoning about them — the kind of reasoning that an agent must do in devising a plan to achieve its goals. Within such a theory, it should be possible to represent what an agent knows about a situation and then reason about whether he is able to achieve various goals by doing various actions in the situation.

Most existing planners have completely ignored the need to reason about what agents know and need to know by unrealistically assuming that agents always have perfect knowledge of the domain under consideration. But in the past decade, Moore (1985; 1980), Morgenstern (1987; 1988), and other researchers have proposed theories of knowledge and action that do address this need. These theories do provide reasonable answers to the question of what the relationship between knowledge and action is and have considerably advanced our understanding of the issues involved in reasoning about that relationship. But many problems remain.

This thesis deals with one aspect of the relationship between knowledge and action that has been neglected and is inadequately handled by these theories: the fact that the knowledge required for action is generally relative to the agent's perspective — that it is generally *indexical* (or relative) knowledge rather than objective knowledge. For example,

- if a robot knows the relative position of an object he can go and pick it up — he need not know what his or the object's absolute positions are
- one can soft-boil an egg using a timer without knowing what time it is.

In these cases, the agent has very incomplete knowledge of his situation, at least as far as objective facts are concerned, but this does not interfere with his ability to achieve his goals by doing actions. More generally, an agent may have sufficient knowledge to be able to achieve his goals even if he does not know

- where he is
- what time it is
- which objects are around him
- where these objects are located (in absolute terms)
- who he is.

This is the case because the knowledge required is indexical knowledge, knowledge about how one is related to things in one's environment or to events in one's history. This should not come as a surprise, since agents act upon their environment from a particular perspective, a particular place and moment in time. The same action done at different places and times has different effects. So it makes sense that the prerequisites of an action should involve knowledge that is relative to this perspective. Similarly, the knowledge supplied by perception is indexical knowledge. For example, by using his sonar, a robot comes to know how far from him an object is (at the current time); he does not learn anything about the object's absolute position.

In some cases where an agent has indexical knowledge of some fact, he also has objective knowledge of the same fact. Then, one can perhaps substitute the latter for the former in a specification of the knowledge prerequisites or effects of an action. Take, for example, a situation where at 10 a.m. December 7, 1990, Rob the robot knows that there is currently a pop can one meter in front of him; if Rob knows who he is (i.e., knows of someone that he, Rob, is that person) and knows what time it is (i.e., knows of some time t that t is the present time), then he must also know of Rob and of 10 a.m. December 7, 1990 that there is a pop can one meter in front of that person at that time (*de re*).[1] In this case, it would not be too inaccurate to say that Rob must have the latter knowledge if he is to be able to achieve the goal of holding a pop can by extending his arm one meter and grasping. In such situations, we can use the latter objective form to represent the former relative form. But in cases where the agent does not know what time it is or who he is, the relativity of knowledge cannot be eliminated. For instance, in the egg-boiling example discussed earlier, at some point the agent knows from the timer that the egg started boiling 5 minutes earlier (and thus that he should be removing it from boiling); since the agent does not know what time it is, he does not know of the current time t that the egg started boiling 5 minutes before t — there is no objective way in which the agent knows the requisite fact. So a specification of ability that would require objective knowledge would not do justice to the situation. Cases where an agent does not know who he is seem more unusual, but they can nonetheless arise.[2]

[1] *De re* knowledge attributions assert the existence of some kind of epistemic relation between the knower and some individual or thing. The intuition behind such attributions is that they apply to cases where the agent's knowledge is sufficiently precise to pick up a particular individual or thing, as opposed to being about whoever satisfies a description. The latter cases are said to involve mere *de dicto* knowledge. For example, if John is told that someone among his co-workers has been selected to be the new manager, but is not told whom, he may then come to believe that the new manager (whoever he is) must be happy — a *de dicto* rather than *de re* belief. If he is later told that the new manager is Paul, he will then come to have a *de re* belief of Paul that he must be happy. Note that we take the *de re*–*de dicto* distinction to be orthogonal to the objective–indexical distinction; the latter has to do with whether the information known refers to the agent's perspective or not. This is discussed further in section 3.1.

[2] Moreover from a theoretical standpoint, one could argue that in the case of the robot above, it is only because he knows that he is a and that the current time is t that his knowing of a and t that there is an

Existing theories of the relationship between knowledge and action cannot handle these cases properly as they do not accommodate the distinction between indexical knowledge and objective knowledge. They impose unnecessarily strong knowledge requirements upon agents before sanctioning their ability; they tend to require *de re* knowledge when mere indexical knowledge would be sufficient. They also cannot represent accurately the effects of action upon knowledge. We need an account of ability that squares with the fact that agents may be missing very basic facts about his objective situation and that of the things around him and still be capable of achieving his goals. The reason we care about this is that this form of incomplete knowledge is the norm rather than the exception for agents in the world.[3]

1.2 Methodology

The goal of this thesis is to develop a theory of knowledge and action that handles the distinction between indexical and objective knowledge and accounts for indexical knowledge prerequisites and effects of action. The theory should capture the kind of reasoning that an agent must do in devising a plan that achieves his goals.

The approach taken is to develop a quantified modal logic that embodies the theory. The logic includes primitives that formalize notions of knowledge, time, action, and historical necessity — the latter notion is used to capture that of a course of action being physically possible for an agent. The notion of ability is formalized as a derived operator (i.e., defined in terms of the notions mentioned above). The epistemic and temporal parts of the logic adequately support the representation of indexical knowledge, keeping it distinct from *de re* knowledge. We provide a formal semantics for the knowledge operator that handles indexicality; it is a simple and natural extension of the standard possible-world semantic scheme for the logic of knowledge (Hintikka 1969; Halpern and Moses 1985).

The specification of the theory as a logical system has several advantages. First, it

object one meter in front of a at time t implies that he is able to achieve his goal (a similar argument is made by Perry (1979)). An account of knowledge and action that ignored this would seem to miss an important part of the explanation for why the agent is able to achieve his goal.

[3]In fact, we think that cases where one only has indexical knowledge are generally more fundamental than cases where one has *de re* knowledge. Having *de re* knowledge of some entity generally presupposes having previously had indexical knowledge of that entity. Moreover, it seems that even very simple autonomous cognitive agents (e.g., robot ants) would need indexical knowledge, while the kind of systems that do not have much indexical knowledge (e.g., large data bases) depend on other agents who do for their usefulness.

ensures that the specification of the theory is precise. Secondly, the provision of a formal semantics clarifies the meaning of the theory's claims and the ontological commitments involved; this makes it easier to judge whether these claims are correct or at least reasonable. Thirdly and perhaps most importantly, the properties of the notions formalized can be studied within the logic; these properties are reflected in which sentences are valid in the logic. For example, if the theory ensures that whenever an agent knows that φ, he must know that he knows that φ, then the sentence that expresses this in the logic must be valid. Fourthly, the specification of the theory as a logic makes it easier to prove various results about it, for instance, results concerning the decidability or tractability of the validity problem for certain classes of sentences; often, established results concerning similar logics can be used or adapted. Finally, one can use such a logic to obtain a precise specification of a system that implements some of the reasoning formalized by the logic. Following Levesque (1984a), it has become common in knowledge representation to specify the behavior of a knowledge representation module in terms of a logic of belief. From our logic, it would be easy to obtain a specification for a planning program — a program that given a goal and a representation of what is known about a situation, generates a plan to achieve the goal. Note that our methodology differs little from that used by Moore (1985; 1980), Morgenstern (1987; 1988), and others who have previously worked on theories of knowledge and action (although the kinds of logical systems used have been different).

Of course, this methodology does not directly address the issue of how one could develop a planning program that handles the kind of situations covered by this theory. But it is generally agreed that planning programs that are truly general and extensible need to be based upon a satisfactory theory of the kind of reasoning required. Our theory attempts to provide this kind of foundation.

A major part of this thesis is involved with the formalization of various domains within the logical system proposed. This is essential to ensure that the theory is truly useful and that it handles adequately the kind of situations that it is designed for. Moreover, the formalization of actions involves many decisions that have a crucial effect upon what knowledge is required from an agent, but are not settled by the theory; for example, decisions concerning the kind of parameters an action should take. The examples we develop show how actions can be specified so as to avoid making excessive requirement upon the knowledge of agents, and how such specifications can be used to prove that an agent is able to achieve a goal by doing an action if he knows certain facts. We direct a significant part of our formalization efforts at a robotics domain, since this kind of application provides the most intuitive examples of situations where indexical knowledge is sufficient for ability. But we also formalize other domains involving temporal knowledge, knowledge of the phone system, and knowledge of one's position in a data structure. The examples involved show that the notion of indexical knowledge is more abstract than one might first imagine. On the basis of evidence provided by the temporal examples, we provide an argument to the effect that the distinction between indexical and objective knowledge accommodated by our framework is of practical interest and that it cannot be handled within previous theories.

1.3 Outline of the Thesis

In chapter 2, we review previous work in artificial intelligence, logic, and philosophy that is directly relevant to our objective of developing an adequate formalization of indexical knowledge and its relation to action. We focus on previous attempts to formalize the relationship between knowledge and action, previous work on indexical knowledge, and recent work on reactive planning and situated action, where the important role played by indexicality in action has been recognized.

In chapter 3, we develop the logical system that forms the basis of our framework, a system that formalizes notions of knowledge, time, action, and ability. The epistemic and temporal part of the logic adequately support the representation of indexical knowledge, keeping it distinct from *de re* knowledge. The complete system permits an adequate characterization of indexical knowledge prerequisites and effects of actions in a wide range of circumstances. The chapter includes a thorough introduction to the logic, a formal specification of its syntax and semantics, a set of useful definitions, a complete review of the properties of the notions formalized, a formalization of ability as a derived notion, and finally a discussion of the contributions and limitations of the system.

In the next two chapters, we apply the logic developed in chapter 3 to the formalization of various domains where agents have goals to achieve and certain repertories of actions to do so. We show how the resulting theory allows us to prove that the agent is able to achieve certain goals if he knows certain facts. We also show how the actions involved should be formalized so as to avoid making excessive requirements upon the knowledge of agents —

so that only indexical knowledge, as opposed to *de re* knowledge, is required when that is sufficient.

Chapter 4 examines applications of the framework in the robotics domain, where indexicality plays a particularly important role. We show how actions can be formalized, given that perception yields indexical knowledge, and that ability to act upon an object does not require *de re* knowledge of the object or its absolute position. We also show how indexical knowledge and objective knowledge can be related in our framework to deal with the use of maps for navigation. We discuss the representational issues that arise, which have general relevance to the formalization of actions with indexical knowledge prerequisites or effects. Finally, we discuss problems that arise in handling some higher-level actions with object parameters.

Chapter 5 shows that the distinction between indexical and *de re* knowledge is quite abstract and not tied to the notion of physical space; it shows that our framework can be usefully applied in several other domains. We first discuss an example involving an agent who wants to make a phone call; we show how this can be handled in a way that reflects the fact that the agent need not always know what his area-code is. We next look at a very abstract domain, that of data structure search and manipulation; we show how indexical knowledge is also relevant in this case. We then look at three cases where temporal knowledge is required for ability. We find that the distinctions between various kinds of knowledge that proved useful in spatial contexts are similarly useful in dealing with ability in temporal contexts. Our ability to handle all of these kinds of situation provides strong evidence for the adequacy of our theory. We then discuss problems with the notion of *de re* knowledge and its role in action. We conclude by arguing that the applications discussed, in particular the ones involving temporal knowledge, provide convincing evidence that the distinction between indexical and objective knowledge supported by our framework has substantial practical value and that it cannot be done justice within existing accounts of ability.

Chapter 6 summarizes the main results of the thesis and suggests various directions for further research.

For a quick overview of the thesis, one may want to go directly to the introduction to the logic in section 3.1, then look at the definitions in section 3.4 and the formalization of ability in section 3.6, then go through some of the examples in chapters 4 and 5, and conclude with the summary of the contributions of the thesis in section 6.1. Also going through sections 3.5.2 and 3.5.7 will give a much better understanding of the properties of knowledge in the logic.

Chapter 2

Background

In this chapter, we review previous work in artificial intelligence, logic, and philosophy that is directly relevant to our goal of developing a theory of knowledge and action that handles the distinction between indexical and objective knowledge and accounts for indexical knowledge prerequisites and effects of action. This thesis straddles the research areas of reasoning about knowledge, and reasoning about action and planning. The volume of work that has been produced in each of these areas is very large, so we will not attempt to do a comprehensive review of each. Good surveys have been produced by McArthur (1988) for the area of reasoning about knowledge and by Georgeff (1987) for reasoning about action and planning. The review in this chapter will focus on previous attempts to formalize the relationship between knowledge and action, previous philosophical work on indexical knowledge, and recent work on reactive planning and situated action, where the important role played by indexicality in action has been recognized.

2.1 Knowledge and Action

As argued in section 1.1, agents need knowledge in order to be able to achieve their goals by doing actions; for example, to be able to open a safe, one must know its combination; to be able to bake a soufflé, one must know how to prepare it. Also, lacking some required piece of information need not be the end of one's hopes of achieving a goal, as one can also acquire knowledge by doing actions; for example, by looking the method up in a recipe book, one can find out how to bake a soufflé. Agents can thus plan their way around their ignorance. As mentioned earlier, to do this, agents must reason about:

- the effects various actions would have upon their knowledge
- the knowledge prerequisites of the actions they might do — the conditions under which they are able to achieve goals by doing actions.

A good theory of the relationship between knowledge and action must formalize these notions and how they relate, to an extent sufficient for capturing this kind of reasoning. Within such a theory, it should be possible to prove that if an agent knows certain facts, then he must be able to achieve a goal by doing an action.

Most of the work aimed at developing relatively efficient planning programs, starting from STRIPS (Fikes and Nilsson 1971) and continuing with numerous successors (Sacerdoti 1977; Chapman 1987), has simply sidesteped these issues by assuming that agents have perfect knowledge of the domain under consideration.[1]

But over the past decade, significant progress has been made in the development of a satisfactory theory of the relationship between knowledge and action. In this section, we will review the various accounts that have been proposed.

2.1.1 Moore

The need for agents to reason about their ability to achieve goals by doing actions was first recognized by McCarthy and Hayes (1979). Their solution to the problem involved adding axioms that stated explicitly what the knowledge prerequisites of actions are. Moore (1980; 1985) was the first researcher to propose a theory of the relationship between knowledge and action that accounted for the knowledge prerequisites of action from general principles. The theory he developed is remarkable in its elegance and explanatory power. Subsequent accounts have borrowed extensively from the ideas it advances. The theory developed in this thesis is more closely related to Moore's than to any other. So we will go over it in more detail than we will do for other accounts.

Moore's first contribution was to develop a logical framework in which one can talk about

[1]This is a bit of an overstatement. As Lifschitz (1987) has shown, STRIPS can handle some cases where the world model it maintains includes disjunctive and existentially quantified formulas. But the framework does not distinguish between what is known and what is true; so it cannot deal with knowledge acquisition actions. For example, consider a robot who can determine the shape of the block he is facing by doing the "sense" action; what formulas would the STRIPS operator corresponding to "sense" add to the world model? None that is expressible without talking about what is known! Yet it is clear that some goals cannot be achieved without such actions being done (e.g., holding a pyramid, when both pyramids and cubes are present).

knowledge, action, and the interaction between the two. Simplifying a little, we can describe this framework as the combination of first-order dynamic logic (a modal logic of action; see Goldblatt (1987)) with the S4 modal logic of knowledge (Hintikka 1962; Halpern and Moses 1985).[2] Both of these logics have possible-world semantics; Moore obtains a semantics for the combined system by interpreting the worlds as possible states (i.e., snapshots of the world at a given moment) and taking the accessibility relations that are used to interpret the modal operators "know" and "after action δ" to both hold over these possible states.[3] Strictly speaking, Moore's framework is not the modal logic just described, but rather the specification in ordinary first-order logic of the semantics of this modal logic, that is, of a truth predicate for it. By taking this approach, Moore can take advantage of the conciseness of the modal object language, while retaining the ability to prove facts about the theory in the metalanguage with an ordinary theorem prover. (Much like the situation calculus, the resulting system is not efficient enough for practical applications, but can be used to prove small examples.)

Within this framework, Moore can specify the effects of an informative action upon the agent's knowledge in a straightforward manner, that is, by adding an axiom to the effect that after doing the action, the agent has that information. For example, in a block-world domain, one might state that after doing the action of looking, the robot would know whether there is a block in front of him. Note that Moore assumes that after doing any action, an agent always knows that he has just done this action and remembers whatever he knew before doing it. As we will see in section 3.1 this assumption is pretty much necessary if agents are to be judged able to achieve goals by doing actions involving more than a single step. This is also axiomatized in a straightforward way.

The major part of Moore's efforts was directed at obtaining a satisfactory specification of the knowledge prerequisites of actions, that is, of the conditions under which *an agent is able to achieve a goal by doing an action.* Note that ability is understood as requiring not just that it be physically possible for the agent to achieve the goal by doing the action, but also that he have the necessary knowledge to do so. The account he produces does

[2]Moore actually presents the framework as being based on the situation calculus (McCarthy and Hayes 1979) rather than dynamic logic. But since the latter is essentially a generalization of the former and since Moore uses the complex action forms of dynamic logic, it seems just as accurate to describe it as we do.

[3]In the logic of knowledge case, the worlds are usually viewed as modeling possible complete world histories, and in the dynamic logic case, they are normally taken to model possible states of a computation.

not involve an explicit axiomatization of the knowledge prerequisites of each action; his specification of the conditions under which an agent is able to achieve a goal by doing an action is based on general principles.

The account involves a modal operator CAN, where CAN($\theta^a, \delta, \varphi$) is intended to hold if and only if agent θ^a is able to achieve the goal φ by doing action δ. A specification of the truth conditions of such formulas is given. The axioms involved make the ability of an agent to achieve a goal by doing an action dependent upon his having the necessary knowledge. The conditions imposed yield an ability operator that corresponds fairly well to the intuitive notion of ability.

The formalization makes use of the fact that actions are represented by a type of term and that, since we are in a modal framework, we can distinguish between the action description or concept corresponding to an action term and the sequence of primitive actions denoted by the term in a particular possible world. In particular, it makes sense to talk about an agent knowing what an action description is or an agent having *de re* knowledge of an action.[4] This notion is important because an agent may well know that an action described in a given way achieves his goal without being able to do it because he does not know what primitive actions the description denotes. For example, one may well know that cooking beef bourguignon would impress one's guests without knowing how to do it, that is, without knowing what primitive actions "cooking beef bourguignon" really stands for. This also applies to actions that are instances of general procedures: if one does not know the combination of a safe, then one does not know what dialing the safe's combination amounts to (even though one knows the procedure for dialing any given combination).

The possible-world semantics' account of *de re* knowledge goes as follows: in a world w, an agent a knows of an entity x that it is φ if and only if x is φ in all worlds that are compatible with what a knows in w (i.e., in all worlds that are knowledge-accessible for a in w). Following Hintikka (1962), knowing who/what θ is is taken to be equivalent to knowing of some x that it is θ. This does not however say much about what is actually required for an agent to have *de re* knowledge; the question is recast into that of what is required for the same individual to exist in distinct epistemically possible worlds. Actually, Moore uses

[4]Moore states that "knowing what action is referred to by an action description means having a rigid designator for the action described ... a rigid designator for an action must be an *executable description* of the action".

a substitutional interpretation of quantifiers, but since rigid designators are substituted for variables, this has little effect on the issue. There has been much philosophical debate on this question but the common answer in AI circles has been that knowing who someone is requires having a *standard name* for that person and similarly for knowing-what (Konolige 1986; Levesque 1984a).[5] Moore (1985) is not very clear as to whether he actually holds this view, but his assertion that "in describing standard identifiers we assumed that everyone knew what they referred to" seems to indicate that this is the case. The account works well for some kinds of objects (e.g. phones) and some tasks (e.g. question answering) but requires some serious stretching in other contexts; for example, what are standard names for a mobile vision-equipped robot?

Let us now go over Moore's specification of the truth conditions of CAN formulas. The specification is recursive; each axiom handles a particular form of action expression.[6] When the action is simple, that is, contains no control structures, the specification says that an agent is able to achieve a goal by doing it if and only if he knows what the action is, knows of himself (*de re*) that it is physically possible for him to do it next, and knows of himself that if he does it next, the goal will necessarily hold afterwards. For a sequentially composed action, the specification says that an agent can achieve a goal by doing it if and only if by doing the first action, he can achieve the goal of being able to achieve the original goal by doing the second action. For a conditional action, one is said to be able to achieve a goal by doing it if and only if either one knows that the condition holds and is able to achieve the goal by doing the action described by the "then" case, or one knows that the condition doesn't hold and is able to achieve the goal by doing the action described by the "else" case. Finally, one is said to be able to achieve a goal by doing the action "`while` φ `do` δ" if and only if one is able to achieve it by doing the action "`if` φ `then` (δ ; `while` φ `do` δ) `else skip`". "`skip`" is the empty action; one is said to be able to achieve a goal by doing it if and only if one knows that the goal presently holds.

Note that the specification captures the fact that the agent need not initially know what all the steps that he would need to do to execute a complex plan are; all that is

[5]This idea was originally proposed by Kaplan (1969). We think that formal study of the relationship between knowledge and action holds potential for improving our understanding of this and other epistemological questions.

[6]The version described here is the one stated in (Moore 1980); the version presented in (Moore 1985) is much more concise, but less explicit about what the truth conditions are when the action is a conditional or a while-loop.

required is that he know that he would know what to do next at each step of the plan. Also note how the requirement that the agent know what the initial action is eliminates the need for an explicit specification of the knowledge prerequisites of actions; in particular, if an action is an instance of a general procedure and the procedure is known (formally, an epistemically rigid function), then one knows what the action is if and only if one knows what the arguments to the procedure stand for. Notice too that the agent is required to know who he is. More generally, note that the only way to talk about time is through the dynamic logic operators. Statements that are not in the scope of a dynamic logic operator may be thought to be about the current time.

It should be clear why Moore's theory has been so well received. The theory handles many cases very well, the logical framework he proposes is relatively simple, and his account of ability involves general principles rather than explicit specification of each particular case. Let us now discuss the limitations of the theory. Since this work is concerned with indexicality, we will start with this issue. As argued in section 1.1, it is important for a theory of knowledge and action to capture the facts that informative actions generally yield indexical knowledge and that indexical knowledge is generally sufficient for ability to achieve goals by doing actions. We mentioned the example of a robot who is able to go and pick up an object because he knows its relative position; the robot need not know where he is or what the absolute position of the object is. We also gave the example of an agent who can soft-boil an egg with a timer in spite of the fact that he does not know what time it is. The agent can have very incomplete knowledge of his situation as far as objective facts are concerned, but still be able to achieve goals by doing actions. Moore's work simply does not address this issue; whatever role there might be for indexical knowledge in the examples he formalizes is ignored. Moreover, his framework does not accommodate the distinction between indexical and objective knowledge. Knowledge about absolute times is not handled. Thus for instance, it is not possible to formalize the fact that after looking at a clock, one knows what time it is. Knowledge that is indexical with respect to the agent is also not handled. Due to this, the account must require that an agent know who he is (know a standard name for himself, something clearly objective) in order to be judged able to achieve a goal by doing an action. But this is clearly unnecessary. What need would very simple agents (e.g., robot ants) have for knowing who they are? As pointed out in section 1.1, this also limits the value of the account as an explanation for why agents are able to act. Because his framework does not support the distinction between indexical and objective knowledge and because he does not recognize the issue, Moore tends to formalize his example situations in a way that imposes unnecessarily strong knowledge requirements upon agents before they are judged able to achieve their goals; he tends to require objective *de re* knowledge when mere indexical knowledge would be sufficient. As mentioned before, the reason why we care about this is that lack of objective knowledge is the norm rather than the exception for agents that live in the world.

Going beyond the issues of particular concern to this thesis, perhaps the most serious deficiency with Moore's theory is the relatively narrow range of actions it handles; it does not deal with concurrent actions, plans involving several agents, continuous processes, etc. Even within its limited range of application, the formalization has a few problems. Its specification of the conditions under which an agent is able to achieve a goal by doing a while-loop action is inadequate. The problem is that the conditions imposed in this case are too weak. So an assertion of ability to achieve a goal by doing a while-loop action can be true in a semantic structure in spite of the fact that the loop never terminates (the loop condition never becomes false) and the goal is never achieved. This leaves the ability of agents to achieve goals only *partly* determined by their knowledge of the situation; thus ability is ontologically primitive in this formalization (just like knowledge or time); but this is clearly incorrect. Note that as a theory of the truth conditions of ability formulas — the way Moore specifies it — this is merely too weak, in that it has undesirable models; but if one were to take it as a truth definition for ability formulas, this definition would be circular.

2.1.2 Other Work

One of the most unrealistic features of Moore's theory is that the knowledge that agents have is closed under logical consequence. Konolige (1982) has proposed a theory of knowledge and action based on an account of knowledge as a syntactic predicate, which avoids this defect. In contrast to the modal approach, the syntactic approach treats knowledge as an ordinary predicate that takes a string argument; the framework used is ordinary first-order logic augmented with quotation. Konolige avoids the consequential closure problem by ascribing an incomplete set of inference rules to agents.

The syntactic approach also yields a more expressive logic than the modal one. One can make very weak statements like "John knows something that Paul doesn't know." But this expressiveness leads to potential problems with paradoxes such as "This sentence is known to be false"; systems based on the syntactic approach must find a way to block the derivation of such paradoxes. Konolige handles this problem by adapting Tarski's approach to the formalization of a language that includes its own truth predicate; this involves a hierarchy of languages and truth (knowledge) predicates over these languages; the truth (knowledge) predicate of a given level can be applied only to sentences from a lower level, which prevents the paradoxes from being expressible.

While syntactic approaches do avoid defects like consequential closure, it can be argued that their model of knowledge is now too fine-grained (Levesque 1984b). For instance, it is far from clear that a belief that φ and φ' is any different from a belief that φ' and φ. Much progress has been made in recent years in the development of non-syntactic logics of knowledge that avoid the consequential closure problem (Lakemeyer 1990).

Konolige's treatment of knowledge prerequisites and effects of actions is essentially a recasting of Moore's account into his framework. Only the same restricted class of actions is handled.

A theory that significantly extends this coverage has been developed by Morgenstern (1987; 1988). Her account of ability handles concurrent actions and plans involving multiple agents. Her treatment of the simpler cases is very similar to Moore's; she retains the key idea that an agent knows how to do an instance of a procedure (parametrized action) if he knows the standard names of the parameters and knows how to do the procedure.

Morgenstern's account of knowledge is also syntactic, but differs significantly from Konolige's. She does not use the Tarskian approach to the paradoxes and there is a single truth predicate for all the languages in the hierarchy.

Note that knowledge is, in fact, closed under logical consequence in her account. Morgenstern's argumentation against modal logics of knowledge is not based on the consequential closure problem but on the claim that the logics cannot express the very weak knowledge requirements involved in multi-agent planning. For example, if Paul does not know how to make sushi, but knows that his friend John knows how (i.e., knows of some action that it yields sushi), he is then able to prepare sushi by first asking John how it is done, and then doing whatever he has been told to do. However, recent work by Nunes (1990) undermines this argument; Nunes shows that the notion of "being able to achieve a goal (by doing some action)" can in fact be formalized in a modal framework.

Note also that agents need not know all valid sentences in Morgenstern's system (i.e., she does not have the knowledge generalization rule). She claims that this is a highly desirable feature in that it allows agents to differ in the amount of procedural knowledge they have. But we fail to see why domain axioms need to be treated in the same way as logical axioms; one could simply look at what the consequences of the domain axioms are according to the logic.

Neither Konolige's nor Morgenstern's work addresses the issue of indexical knowledge; their frameworks do not accommodate the distinction between indexical and objective knowledge and they do not formalize or discuss cases where the distinction is important.

The important role of indexical knowledge with respect to action has been pointed out by Haas (1986), as part of a thorough critique of Moore's theory. He gives an informal sketch of how one might take indexical knowledge into account in a specification of ability in the context of a syntactic account of belief; but he does not formalize his proposals.

Finally, let us point out the work of Davis (1988; 1989b; 1989a), who formalizes the effects of perception upon knowledge within a framework similar to Moore's. He treats "perceiving" as a propositional attitude rather than as an action. He looks at how inferences can be made about agents' ignorance of events outside their perceptual range.

2.2 Philosophical Theories of De Se Attitudes

Most of the work on the semantics of indexical knowledge and other indexical attitudes has been done by philosophers. This section reviews this work. At the end of the section, we briefly discuss a few AI references on the topic.

2.2.1 Castañeda

In the mid-60's Castañeda (1968) published several influential papers on the logic of indexical or *de se* propositional attitudes and the various problems it raises. He starts out by claiming that the pronoun 'he (himself)', to be abbreviated by 'he*', used in third-person statements that attribute self-knowledge (self-belief) to others (e.g., 'The editor of Soul knows that he* is a millionaire'), is not a true indexical, like 'I', 'you', or 'now'; rather, such reports attribute to the agent an attitude towards an object that the agent would voice by asserting a sentence containing the indexical 'I' (e.g., 'I am a millionaire'). For this reason, he calls 'he*' a quasi-indicator (he uses 'indicator' for 'indexical').[7]

Castañeda also points out that iterated attitude reports containing occurrences of 'he*' must be disambiguated by specifying the antecedents of these occurrences. For example, if the antecedent of 'he*' in an utterance of 'Paul knows that John knows that he* is a millionaire' is 'John', then the statement attributes to Paul the knowledge that John has some self-knowledge; if on the other hand the antecedent of 'he*' is 'Paul', then the statement attributes self-knowledge to Paul. Castañeda argues convincingly that occurrences of 'he*' can be moved out of nested attitude scopes to appear immediately within the scope of the attitude operator whose subject is their antecedent. For example, a statement of 'Paul knows that John knows that he* is a millionaire' where the antecedent of 'he*' is 'Paul' can be rewritten as 'Paul knows that $\exists x(x = \text{he}* \wedge$ John knows that x is a millionaire).' But he also claims that the remaining occurrences of the quasi-indicator cannot be eliminated. This contrasts with Hintikka's proposal that *de se* knowledge be analyzed in terms of knowing who — for example, that 'John knows that he* is a millionaire' be analyzed in terms of '$\exists x(x = \text{John} \wedge$ John knows that x is a millionaire).' Castañeda argues that this proposal is misguided, since it is easy to imagine situations where an agent would be said to know who some individual is in spite of his failure to realize that he* is that individual; for example, if John were the world's foremost authority on a certain war hero and yet failed to realize that he* is that war hero.

The main drawback of Castañeda's early work is that it does not provide a semantic account that would support the various logical principles advanced. This would seem to leave the correctness and even consistency of these principles very much in doubt — experience with other aspects of the logic of attitudes has shown that intuitions often lead to incoherent accounts. So we will immediately move on to theories that include proposals on the semantics of *de se* attitudes.[8]

[7]Not everyone agrees with this claim. Perry (1979) and later Barwise and Perry (1983) claim that there are really no such things as *de se* attitude reports; attitude reports containing 'he*' are merely *de re* reports from which the hearer typically draws the pragmatic inference that the agent has a *de se* attitude. Since this work is not really concerned with the semantics of natural language attitude reports, the issue is of peripheral importance to us. But note that in the absence of counter indication, any use of 'he*' in the following discussion will be intended as an attribution of a *de se* attitude.

[8]Castañeda later attempted to provide a semantic account for *de se* attitudes within phenomenalist framework. Due to differences in approach and lack of time, we have not attempted to study and discuss his theory.

2.2.2 Perry

The standard view of attitudes such as belief and knowledge is that they are relations between an agent and a proposition (at a time), where a proposition is some kind of object that has a truth-value that depends only on how the world is; in particular, it doesn't depend on time, place, or person.[9] Classical semantic accounts of attitude reports based on possible worlds seem consistent with this view.

Perry (1979; 1977) argues that if one is interested in characterizing mental states in a way that supports the explanation and prediction of behavior, that is, a way that is consistent with them playing causal roles in determining action, then this characterization is inadequate. His argument is based on instances of *de se* belief, that is, beliefs about where one is, who one is, what time it is (Perry calls them "locating beliefs"). Note that, as Lewis (1979) points out, it makes sense to consider attitudes about the current time to be *de se* if one takes the subject of an attitude to be a person-state (i.e., a person at a time) rather than a time-extended individual. In one example, Perry (1979) describes the case of a shopper who, noticing a trail of sugar on the supermarket floor, follows it with the intention of advising the culprit that he was making a mess, unaware that he himself was that culprit. Eventually, the shopper realizes that he himself is making the mess and thus ceases following the trail and proceeds to rearrange the torn sack of sugar. The problem for the standard view with instances of *de se* belief, such as this, is that the sentence that the agent would use in sincerely describing what he believes, 'I am making a mess', does not identify a specific proposition; it identifies different propositions in different contexts. In order to sustain the standard view, one must argue that the use of the indexical 'I' is really a communicative shortcut, and what the shopper really believes is a proposition of the form 'α is making a mess', where α is a concept that uniquely fits him (in all contexts).[10] But as Perry points out, this is insufficient because 'the shopper believes that θ is making a mess', where θ is a term that expresses concept α, would not be an adequate explanation of why the shopper straightened the sack unless one were to also grant 'the shopper believes that he* is θ'. Moreover it is unnecessary because 'the shopper believes that he* is a making a mess' explains his behavior even if all the concepts he has of himself don't actually fit him

[9]This view is associated with an emphasis on *de dicto* attitudes and either a downplaying of *de re* attitudes or the assumption that the latter can be reduced to the former.

[10]Ignore for the moment that people having such concepts in all cases is highly unlikely.

(he may think he's Napoleon in the army stocks). The problem is that believing that α is making a mess does not characterize the state in a way that is adequate for explaining action.

Perry goes on to discuss whether it might be possible to resolve the problem by explaining *de se* attitudes in terms of *de re* attitudes. This seems reasonable because initial observations clearly indicate that having a *de se* attitude implies also having a *de re* attitude. Moreover, the problems raised by *de re* attitudes for the standard view appear similar to that raised by *de se* attitudes: the sentential argument of the attitude report seems to lack a concept needed to obtain a complete proposition. One approach to *de re* attitudes attempts to explain them in terms of *de dicto* attitudes, the idea being that an agent has a *de re* attitude of some object if he has a *de dicto* attitude involving a concept that uniquely fits the object and where some additional relation holds between the agent, the concept, and the object (e.g., being a vivid name (Kaplan 1969) or denoting the object in all accessible worlds (Hintikka 1962; Hintikka 1969)). But if Perry's earlier argument is right, then such approaches offer no hope for a solution to the *de se* problem because they merely augment the requirements.

Another view of *de re* attitudes is that they are irreducible and that they relate the agent to a new kind of proposition composed of a property and the object itself (rather than a concept of the object)—call them singular propositions. But Perry claims that even this deviation from the standard line does not help in solving the problem of *de se* attitudes. More must be involved in having a *de se* attitude than just having a *de re* attitude of oneself. For example, the shopper may see the culprit in a mirror without realizing that he is seeing himself. His behavior is likely to be different in each case. According to Perry, singular propositions cannot help in characterizing this additional feature of *de se* attitudes because they do not have the requisite indexical character.

Perry proposes to resolve the problem in the following way. What we seem to be doing when we explain or predict behavior is to classify agents according to the sentences containing indexicals that they would sincerely utter. Abstracting away from language, this seems to amount to classifying agents in terms of functions from contexts to propositions—what Perry calls relativized propositions and Kaplan (1977) calls characters (of sentences). It is attitude states thus classified that enter into intentional theories of behavior and not properties such as believing that such and such a proposition (whether standard, singular, or relativized) is true. There is a systematic relation between the attitude states that an agent is in and the propositions he has attitudes towards, but no isomorphism. The attitude state together with the context determine what proposition the agent has an attitude towards. A classification of agents in terms of what attitudes they have towards what propositions is still useful. "For instance, usually, when a believer moves from context to context, his beliefs states adjust to preserve beliefs held." But it is not the classification used in intentional theories of behavior. Note that Perry rules out attitudes being relations to relativized propositions, because for him having an attitude seems to necessarily involve having an attitude that some object is true, and relativized propositions can only be true with respect to some context, which reintroduces the *de se* problem. This account of *de se* attitudes is a key element of Barwise and Perry's recent comprehensive theory of meaning, situation semantics (Barwise and Perry 1983).

2.2.3 Lewis

Lewis (1979) agrees with Perry that *de se* attitudes raise insurmountable problems for the standard view that attitudes are relations to 'objective' propositions. In keeping with his possibilist position (i.e., his view that possible worlds have the same metaphysical status as the real world), he motivates the required changes in a different way. He proposes that we look at the problem according to the following spatial metaphor: ordinary *de dicto* attitudes locate the agent within 'logical space' — they locate him and his world within some set of possible worlds; *de se* attitudes, on the other hand, locate the agent within ordinary space and time (one may view position in logical space as a fifth dimension). So all attitudes correspond to self-ascription of a property by the agent: that of being located in such and such region of logical space or ordinary space-time. The standard view that attitudes are relations to propositions amounts to restricting the agent to self-ascribe only properties that locate him within logical space; in possible world semantics terms: only properties that map a world into the set of all agents (or agent-time pairs) that exist in this world (if the world is in the given region of logical space) or the empty set of agents (if the world is not in the given region of logical space). From his point of view, this appears to be an arbitrary restriction.

So Lewis proposes that we analyze an attitude as a relation between an agent and a time and a property of individuals and times. This is quite similar to what Perry was proposing;

if Perry's contexts are restricted to agent-time pairs, it becomes equivalent.

2.2.4 A Comparison of Lewis's and Perry's Accounts

There seems to be two major areas of disagreement between Lewis and Perry with respect to these matters. The first seems to be mostly terminological. For Perry, agents still have attitudes toward propositions, even though they only have these attitudes in virtue of being in certain attitude states. The attitude states are what enter into intentional theories of behavior, while the attitudes toward propositions are used to explain agreement in attitudes, determine whether the agent is right or wrong, etc. For Lewis, the behavior-determining role of the attitudes is primary — an attitude is in the head. So he objects to Perry's talk about an agent believing a proposition; a belief is a relation to a property; believing a proposition is not a property of agents; an agent's belief in a proposition is a state of affairs involving both the internal state of the agent and contextual facts. Lewis's arguments have a lot of intuitive appeal. But Perry can also defend his position. Common-sense psychological discourse uses largely the same locutions to talk about both roles of attitudes. From this point of view the choice of which role to associate with 'have an attitude' seems arbitrary. Moreover, *de re* attitude reports leave the description under which the agent has the attitude towards the object unspecified, but we still call them attitudes.[11]

The other area of disagreement between Lewis and Perry is more substantive. Lewis claims that *de re* attitudes can be explained in terms of *de se* attitudes and non-intentional relations holding between the agent and the object of the attitude. Specifically, Lewis proposes that an agent believes of an individual x that it has property P if and only if the agent believes that he* stands in relation R to some unique individual y and that y is P, the agent in fact uniquely bears relation R to x, and R is a relation of acquaintance. Lewis does not give a definition of what a relation of acquaintance is, but says that there has to be an extensive causal dependence of the agent's mental states upon the object and this causal dependence must be of a sort apt for the reliable transmission of information. (Lewis only talks about *de re* belief of people, but I presume that he could extend his account to deal with arbitrary attitudes of arbitrary objects.) Examples of relations of acquaintance

[11]We have also mentioned Perry's claim that no attitude reports are really *de se*; reports containing 'he*' are merely *de re* reports from which the hearer typically draws the pragmatic inference that the agent has a *de se* attitude.

would be those one holds with one's acquaintances, present or absent, with public figures prominent in the news, with famous dead figures prominent in history, with strangers one sees, etc. Identity is a relation of acquaintance, so *de se* attitudes are also *de re*.

In view of this claim, Lewis finds Perry's account unnecessarily general: if *de re* attitudes can be reduced to *de se* attitudes, the only singular propositions that one would ever have an attitude towards would be those that attribute a property to oneself; under these conditions, the explanatory contribution of 'having an attitude towards a singular proposition' in addition to 'being in an attitude state characterized by a property' appears negligible. For singular propositions to play a significant explanatory role, the attitude states must be characterized by functions that map more than just an agent and a time into a proposition; the context mapped must involve individuals other than the agent.

2.2.5 Stalnaker

Stalnaker (1981) rejects the conclusion of Perry and Lewis that *de se* attitudes cannot be accommodated within the standard view that attitudes are relations to 'objective' propositions. He argues on behalf of this position by proposing ways to handle puzzles similar to Perry's messy shopper within the possible-world semantic framework associated with standard modal logics of knowledge, and by pointing out advantages that his account would have over the more complex accounts of Perry and Lewis. We will see that his proposed solutions to the puzzles turn crucially on the fact that, under the version of the metaphysics of possible worlds that he uses, it is considered meaningful to have a possible individual exist in more than one world (independently of any properties he may have).

The main example discussed by Stalnaker (introduced earlier by Perry (1977)) involves Lingens, an amnesiac lost in a library who has read an extensive biography of himself; he knows that the subject of the biography is named 'Lingens' and is the cousin of a spy, but he doesn't know that he* is the subject of the biography and therefore is the cousin of a spy; in fact he has no opinion as to whether he* is or is not Lingens. Stalnaker proposes the following characterization of Lingens's mental state: in all of Lingens's belief alternatives (i.e., in all of the alternative possible worlds compatible with his beliefs), there is an individual i_1 that has read the biography and has a belief state (the belief state under discussion); some of these worlds are like w_1, where i_1 is named 'Lingens', and is the cousin of a spy; but some are like world w_2, where i_1 is not the cousin of a spy, and another possible individual i_2 is named 'Lingens' and is the cousin of a spy. Note that the possible individuals in the extension of the concept of the agent who has read the biography are identified across worlds. Under this characterization, Lingens knows who he* is, but his belief that Lingens is the cousin of a spy is not really a belief of Lingens but a belief about whoever falls under the concept of being the person named 'Lingens' and the subject of the biography (assuming the standard account of 'knowing who' and *de re* belief due to Hintikka (1962)).[12]

Stalnaker notes that his account seems to violate the "independently plausible" semantic view that names, such as 'Lingens', are rigid designators. But he goes on to propose an account of the relationship between the semantics of the sentence 'Lingens is the cousin of a spy' and the belief state of an agent who would sincerely assert such a sentence. Roughly, the sentence corresponds to a function from contexts, which determine the reference of the name 'Lingens', to propositions; but the object of the belief of the agent who is disposed to sincerely assert such a sentence is not the associated function from context to proposition, nor the proposition associated with the sentence in the actual context, but ***the proposition that the sentence expresses a true proposition*** (whatever that proposition is), that is, the set of worlds w such that the sentence if uttered in w would be true in w. This fits with the characterization given earlier of Lingens's belief state. Stalnaker claims that this account is compatible with the view that names are rigid designators and that their reference is determined by some kind of causal chain: "the proposition in question [believed by Lingens] is a function of the propositions determined by, in the various alternative possible situations, the semantic account which assumes that names are rigid designators". I take this to mean that 'Lingens' is rigid with respect to necessity in all the belief alternatives of the agent, but is not rigid with respect to these belief alternatives.

This also fits with other comments of Stalnaker concerning Kripke's arguments for the view that names are rigid designators. He says that Kripke is probably right in claiming that one does not need a criterion of identity to make counterfactual suppositions concerning particular individuals, one can just stipulate that the counterfactual situation involves

[12]Stalnaker mentions that in his view, there is another equally valid characterization of the example: Lingens knows who Lingens is, that is, the same individual is named 'Lingens' in all his belief alternatives, but Lingens does not know who has read the biography and has the belief state in question. In having two equally good characterizations, Stalnaker says that the Lingens example resembles the better known puzzles of *de re* belief (such as Quine's famous Ortcutt example (Quine 1956)). In fact, he says that his account of the example, that is, of *de se* belief, treats it as a special case of a more general problem about *de re* belief.

these individuals. But one cannot stipulate what individuals are in one's belief alternatives because one's ability to make such stipulations presumably depends on the fact that one can have beliefs of these individuals. Stalnaker envisions that an account of *de re* belief would also be based on causal dependencies, but says that "no matter how this kind of answer is filled out, it would be highly implausible to expect that there should always be a nice one-to-one correspondence between certain individuals in the actual world and those who inhabit the possible worlds which define a state of belief. ... In cases where two individuals in a possible situation have equal claim to be identified with an individual in the actual world (or the reverse), how the identification is made may depend more on the context of belief *ascription* that it does on the character of the belief state itself".[13]

Given the complexity of the issues raised by Stalnaker's account of the Lingens example, it is easy to lose track of what is essential to his proposal with respect to *de se* attitudes in general. Let me point out some aspects of his account of Lingens that I believe cannot be essential to a general account. Lingens has learned about the Lingens character by reading his biography, that is, through language. This is what allows Stalnaker to characterize Lingens's belief state based on the various individuals that may have the property of *being named 'Lingens'* — a linguistic property if there is one. Lingens's belief alternatives are all (in part) linguistic contexts for the sentence 'Lingens is the cousin of a spy'. This is also why Stalnaker can describe the situation by saying that Lingens believes that "whichever proposition the sentence expresses, it is a true one". But clearly, one can imagine similar situations where the agent has achieved an attitude about himself through non-linguistic means, for instance, Perry's shopper example. Nor could a sentence that the agent might utter to describe his attitude play the same role in general because the agent may not know any language or at least may not have in mind any linguistic way of expressing his belief at a given time.

Another aspect of Stalnaker's account that seems inconsequential, as far as a general characterization of *de se* attitudes is concerned, is the fact that the belief alternatives of an

[13]Note that Stalnaker does not claim his explanation of the relationship between Lingens's belief state and the sentence 'Lingens is the cousin of a spy' to constitute an adequate account of the semantics of belief reports in general, only that it seems to fit intuitions in the Lingens case; he adds that the proposition denoted by the sentential argument of a belief report appears to be context-dependent and due to this such a general account would likely be more complicated than is generally supposed. This is to be expected: if it were not context-dependent, how could names be 'semantically rigid' while sometimes contributing a non-rigid concept to the proposition believed by the agent, as in the Lingens example?

agent will always contain a representation of the belief itself. In the Lingens example, all of Lingens's belief alternatives contain some individual who has read the biography and has a belief that Lingens is the cousin of a spy. This feature will be present in all cases and does not appear objectionable. But it does not seem to be of any help in capturing the difference between *de se* and non–*de se* attitudes because there might be a lot of other beliefs with different agents in these worlds; the fact that an individual has a belief in an agent's belief alternatives in no way implies that that individual is the agent. I will come back to this later in this section.

Another example discussed by Stalnaker brings forth an important assumption underlying his account: it must make sense to talk about whether a possible individual that exists in one world also exists in another world (without reference to common attributes and behavior), that is, cross-world identity must be possible and must be a primitive — this position is referred to in the philosophical literature as Haecceitism (Kaplan 1975). The example, due to Lewis (1979), involves two gods who are omniscient with respect to non-indexical propositions; there are two mountains in the gods' world and each god wonders which mountain he* lives on. Stalnaker claims to handle the example by having each god's belief alternatives contain two worlds that are identical in the (objective) facts that they support, but where the possible individuals corresponding to the gods are interchanged. Thus, Stalnaker's metaphysics contains more propositions than can be constructed from strictly objective facts.

Stalnaker also discusses examples where an agent does not know what time it is. He points out that for his approach to work in all such cases, "we must be referring to the particular thought token that O'Leary [the agent] is thinking". But he is not clear as to how exactly one should model such situations. It seems that one way to do it would be include times in the ontology; then, the attitude of the agent can be characterized by a set of alternatives analogous to the other examples, but where the alternatives differ in the times at which the agent has the attitude.

2.2.6 A Comparison of Stalnaker's Account with Perry's and Lewis's

Now that we have looked at what Stalnaker was proposing, let's see what arguments he gives as to why one should prefer his account to that of Lewis and Perry, which he appears to take as adequately capturing the distinction between *de se* and non–*de se* attitudes.

Stalnaker (1981, p. 143) says:

> But I think that the identification of such possible situations with centered worlds in Lewis's sense [equivalent to properties of individuals and times], would make it more difficult to give an adequate account of the relations between different states of mind. Specifically, it would make it more difficult to compare past with present beliefs, and more important, to explain the relations between the beliefs of different persons — relations that are essential to a natural explanation of the exchange of information.

It easy to understand why Stalnaker thinks his account allows a simpler explanation of such relations: for him the objects of attitudes are the same kind of entities as the content of statements — propositions; moreover, the objects that Lewis and Perry use to characterize attitude states are relative to the agent and time of the attitude and thus must be adjusted when transferred to the perspective of a different agent and/or time if their content is to be preserved.

Consider how one would explain Lingen's learning that he* is Lingens through being told 'You are Lingens' by a reliable source. Stalnaker can claim that the proposition expressed by this sentence in the given utterance context is one that distinguishes between the possible situations that were compatible with Lingens's beliefs in a way that resolves his doubt. But Lewis's explanation must be more complicated. If he takes the objects of speech acts to be properties, then the person uttering the sentence resolves Lingens doubt by self-ascribing a property which is distinct from the property that would resolve Lingens's doubt if he were to self-ascribe it; the resolution of his doubt must involve an inference on Lingens's part. If Lewis, on the other hand, takes the objects of speech acts to be propositions, then "he must deny that speech is a straightforward expression of thought". Stalnaker would probably apply the same criticisms to Perry's framework, since his account of communication would surely involve a combination of both objects (as this is the case in situation semantics (Barwise and Perry 1983)).

Stalnaker also criticizes Perry's account on the grounds that neither his belief states nor his belief contents (propositions believed) are adequate characterizations of the information content of the statement that resolves Lingens's doubt. The proposition expressed by the statement and believed by both agents cannot do the job because it is the trivial proposition that Lingens is Lingens, which was already believed by Lingens before the statement was made. Perry's propositions are just "too extensional to characterize the information conveyed in acts of communication". But Perry's belief states are also inadequate to characterize information content because they are "too subjective": the belief state of the speaker is different from that of Lingens. I think that if this criticism is valid, it applies just as well to Lewis, since his properties are also "too subjective" and his propositions are not fine-grained enough because he rejects the kind of cross-world identity necessary to obtain Stalnaker's 'fine-grained' propositions.

Should one accept Stalnaker's arguments that his account, which espouses the standard view that attitudes are relations to propositions, shows that such a view is adequate for characterizing *de se* attitudes, and that it leads to simpler accounts of communication, interpersonal agreement, belief retention, etc.? Lewis (1979) anticipates the kind of account that Stalnaker proposes and presents an argument against it. The argument involves the situation described earlier where two gods who are omniscient with respect to non-indexical facts are wondering which mountains they* live on. Remember that Stalnaker's account of the example has the gods' belief alternatives contain two worlds where the same facts hold but where the individuals corresponding to the gods are interchanged. Lewis claims that Stalnaker's characterization of the belief state of a god when he does know that he* lives on a given mountain is inadequate. In this case, Stalnaker would have the god's belief alternatives contain only one of the possible worlds described above, call it w. But, says Lewis, "There are still two mountains in w where he might, for all he knows, be living". For any given proposition, the god knows whether it is true in w; he knows that the proposition that he would express by saying 'I am the god on the tallest mountain' is true; but he doesn't know whether he would be making a true statement by saying it because he doesn't know which proposition the statement would assert.

Stalnaker opposes Lewis's argument by saying that "One cannot just stipulate that the god knows that he is in w, and not in w′, for on the proposed explanation, that amounts to the assumption that he knows which mountain he is on." He points out that Lewis must admit the kind of account he is proposing works for ignorance of the location of others. If one were unclear as to whether Ronald Reagan is now in Washington or California, and one were to learn that he is in fact in Washington, it would be correct to characterize one's belief state as going from a set of alternatives that includes worlds where the individual that corresponds to Reagan is in Washington and worlds where the same individual is in

California, to a set of alternatives that contains only worlds where the individual is in Washington.[14]

Stalnaker's proposals and argumentation are quite seductive. At first, it appears that he may have found a way of distinguishing between *de se* and non–*de se* attitudes without representing the agent's perspective. In the 'Lingens' case, he is able to produce an account within the confines of the 'attitudes as relations to objective propositions' framework because of a seemingly incidental fact: there is one fewer agent in the *de se* case than in the non–*de se* case. It is hard to pinpoint why his account of the 'gods' example seems to work, but harder still to locate a flaw. But deeper analysis reveals what appear to be very serious problems with Stalnaker's proposals.

Firstly, Stalnaker's account of *de se* belief, when joined to the standard treatment of *de re* belief he uses (due to Hintikka (1962)), leads to very counterintuitive results. We have seen that in the Lingens example, Stalnaker's approach produces two characterizations (claimed to be equally valid): one where Lingens knows who he* is but does not know who Lingens is, and another where Lingens knows who Lingens is but does not know who he* is. But Stalnaker cannot have Lingens's beliefs about himself and about the subject of the biography both be *de re* without having Lingens know that he* is Lingens. His characterization of Lingens's ignorance that he* is Lingens depends on the presence among Lingens's belief alternatives of worlds where the subject of the biography and the agent of the belief are different possible individuals. But it seems that there are analogous situations where both beliefs must be *de re* without being *de se*, for example, the situation described earlier where Perry's shopper sees the culprit in a mirror without realizing that he is seeing himself.

It may be claimed that this argument fails to be compelling because it is based on an unexplained notion of *de re* belief and because it has not been shown that Stalnaker's notion of *de re* belief is inadequate in general. But there is another line of criticism that strikes at the heart of Stalnaker's proposal; this line is in fact very similar to Perry's argument (Perry 1979) against the view that attitudes are relations to 'objective' propositions (discussed

[14]Clearly, it is not's the agent's role to know which mountain he lives on in w; his belief alternatives are determined objectively by his belief state. It is Stalnaker's assumption that if the agent does not know which mountain he* lives on, his alternatives must also include w'. I am not implying that Lewis makes this mistake. But his argument is not convincing, because he fails to clearly identify circumstances under which the god may be in a state where his only belief alternative is w without knowing which mountain he* lives on.

earlier in this section).

The argument goes as follows. Suppose that Lingens has an individual concept α that uniquely fits him in all possible worlds (in which he exists, and all contexts) and believes that α is named 'Lingens' and is the cousin of a spy, but does not realize that he* is α.[15] Then under Stalnaker's characterization of belief states, Lingens's state is indistinguishable from that ascribed to him when he believes that he* is named 'Lingens' and is the cousin of a spy, that is, realizes that he* is α; in both cases, there is a possible individual who appears in all of Lingens's belief alternatives and who is named 'Lingens', is the cousin of a spy, and is the agent of a belief in every alternative. The problem is that these belief states would likely lead Lingens to adopt different courses of action, and if the theory is to be adequate for supporting intentional explanations of behavior, then it should characterize these states as different. This appears to confirm the view that the concept of the self is essentially indexical and that this indexicality must be modeled in any adequate theory of belief.

Stalnaker has some possible avenues of defense left. He might try to deny that such a concept could be knowable in principle. Or he might try to argue that whenever an agent believes that α has property P, where α is a concept that uniquely fits him in all possible worlds, he also believes that he* has property P.[16] But I fail to see how such positions could be sustained.[17]

2.2.7 Conclusion

Thus it appears that, even though it might have been nice for Stalnaker's account to work out so that the simple kind of account of communication he sketches could be supported, one will have to live with the additional complexity. The two kinds of classifications of attitude states proposed by Perry, the first in terms of functions from contexts to 'objective' propositions, and the second in terms of 'objective' propositions (or the corresponding

[15]Of course, it is unclear what such a concept would involve, personal identity being a notably tricky notion. But the existence of such individual concepts seems to be generally accepted in possible world semantics.

[16]Note that this would involve more than the kind of idealization that gave us 'logically omniscient' agents. No matter how logically competent the agent would be in the situations described above, he could not deduce that he* is Lingens.

[17]By the way, note that if we replace 'an individual concept α that uniquely fits him in all possible worlds' in the above argument by 'an individual concept α that uniquely fits him in all of his belief alternatives', then the argument becomes essentially the same as the prior argument concerning *de re* belief.

kinds of objects in Lewis's framework), appear to have a distinct and essential role to play in theories of cognition and communication. For Perry, it is wrong to attempt to locate the totality of the information conveyed by a statement in the 'objective' proposition that it expresses, as proposed by Stalnaker. In situation semantics (1983), the meaning of statements is explained in terms of the constraints they impose on the context of utterance and the 'objective' proposition expressed; the hearer's (tacit) knowledge of these constraints is what explains his understanding of statements containing indexicals, where more information is conveyed than is contained in the 'objective' proposition expressed. For example, my answer 'I am Yves Lespérance' to a request of information on my identity conveys more information than that contained in the trivially true 'objective' proposition expressed 'Yves Lespérance is Yves Lespérance'; the hearer acquires the information that the speaker is Yves Lespérance because he knows that the sentence used constrains the context to be such that the speaker believes that he* is named 'Yves Lespérance' under normal conditions.

Before moving on, we should mention some AI work that has dealt with indexical attitudes (there is very little). Smith (1986) has investigated the role of indexicality in our common sense attributions of semantic content to computational agents. He finds that context sensitivity is a much more ubiquitous phenomenon than is commonly thought and argues that it needs to be accounted for if we are to obtain theories of computation that explain how information processing systems fit within their social environment. Similar observations and arguments are made in a more philosophical work by Barwise (1986). In a more practical vein, Rapaport (1986) has developed ways of representing indexical knowledge in a knowledge representation system; no formal semantics is proposed, however. Finally, let us mention very recent work by Grove and Halpern (1989) on the logic of indexical knowledge and its applications in the area of distributed systems.

2.3 Reactive Behavior and Situated Action

In this section, we discuss recent work on *reactive agents*, that is, agents that can react in a timely manner to conditions in their environment that require it. This work is related to ours because it has been recognized that indexical information plays a major role in the production of reactive behavior. It provides additional motivation for our work and points to significant areas of potential application.

The classical AI paradigm with respect to the production of intelligent behavior involves a smart planner that searches for a plan that achieves the agent's goal and a dumb executor that carries out the plan in a mechanical way. In recent years, there has been a definite movement towards exploring alternatives to this paradigm. It is felt that because of the emphasis it places on search for complete plans, classical deliberative planning is too slow for producing reactive behavior (e.g., avoiding collisions with moving objects). Moreover, it is not sufficiently grounded in perception; much of the effort expended on constructing an appropriate plan may be wasted if environmental conditions change in the meantime. The alternative architectures proposed achieve reactivity by emphasizing environment monitoring and the selection of actions appropriate to conditions; the focus is on developing a sophisticated executor.

Agre and Chapman (1987; 1990) have been among the most radical in their reevaluation of the classical paradigm. Their views have been influenced by anthropological theories of action (Agre 1990; Suchman 1987). They emphasize the complexity of the real situations in which action occurs, the uncertainty of the information the agent may have about them, and the need for reactivity. This leads them to argue that the production of most activity does not involve the construction and manipulation of explicit representations of the world; the associated computational costs are just too prohibitive. They say that (Agre and Chapman 1987, p. 268):

> Rather than relying on reasoning to intervene between perception and action, we believe activity mostly derives from very simple sorts of machinery interacting with the immediate situation. This machinery exploits regularities in its interaction with the world to engage in complex, apparently planful activity without requiring explicit models of the world.

The architecture they propose involves peripheral modules for perception and effector control and a central system that mediates between the two. They argue that combinational networks, that is, circuits computing logic functions, can form an adequate central system for most activities; thus in such cases, the central system has no state. They have built various application systems to provide support for their analysis. One of these systems, called Pengi, plays a videogame where one controls a penguin that navigates in a maze, pushes away ice blocks, and confronts malicious "killer bees" (Agre and Chapman 1987).

From our point of view, the most interesting elements of Agre and Chapman's scheme are their notions of indexical-functional entities and aspects.[18] They claim that traditional domain representations involving objective facts such as "(AT BLOCK-213 427 991)" are inappropriate because they do not make reference to the agent's situation or goals. They see the machinery in their networks as registering indexical-functional entities such as "the block I'm pushing" and indexical-functional aspects such as "the bee I intend to clobber is closer to the projectile than I am". These entities and aspects are indexical because they depend on the agent's situation; they are also functional because they depend on his purposes. In Agree and Chapman's scheme, such indexical "representations" play a crucial role in the production of reactive behavior.

The main limitations of Agree and Chapman's work are that they have not proposed any framework for the design of reactive agents or any formal version of their account of activity.

A theory of action that deals with the role of indexical representations in reactive behavior has recently been proposed by Subramanian and Woodfill (1989). Their work is very significant in that it includes an analysis of the efficiency gains associated with the use of indexicals.

To provide a setting for their analysis, Subramanian and Woodfill develop a variant of the situation calculus (McCarthy and Hayes 1979) that allows indexical propositions to be stated. They do this by introducing a distinguished constant `Now`, which is intended to denote the current situation. Other indexical terms such as "the block that is currently nearest to the agent" can then be specified in terms of `Now`. Constraints are also introduced to ensure that every situation has at most one predecessor; the function `Before` is used to refer to a situation's predecessor. Note that this amounts essentially to introducing indexical entities into the ontology.

They then show how the kind of indexical control rules (indexical because they refer to the current situation) that determine what a reactive agent will do next (one-step planning) can be specified in such a framework. This is also done for plan monitoring rules, which are used to deduce what should be the case after an action has been performed. They then show how the rules can be propositionalized . Their analysis of this kind of setting traces the efficiency gains of using indexical representations to the fact that one can thus

[18]This is Agre and Chapman's terminology. There are of course no such things as indexical (or agent-relative) *entities*; but there are indexical (or agent-relative) concepts or notions of things.

reduce the number of entities to be considered in the domain and thus, the theory can be propositionalized without the usual combinatorial explosion. The resulting propositional theory can be reasoned with in exponential time. They also argue that if one assumes that only one action is done at each step and the theory is suitably reformulated, then one can determine which action should be done next in time linear in the number of possible actions. Finally, they claim that if the reformulated theory is then compiled into a logic circuit, then one can decide which action should be done next in constant time. Subramanian and Woodfill could be clearer about the assumptions on which their results depend and about the limitations associated with the kind of theories they analyze. But the analysis is very helpful in clarifying the source of the efficiency gains associated with the use of indexical representations (it is really a case of trading expressiveness for efficiency).

For dealing with the kind of situations that this thesis is particularly concerned with, that is, cases where the agent is ignorant of various aspects of the domain, Subramanian and Woodfill's framework suffers from much the same limitations as the ordinary situation calculus. It forces one to assume that the agent has perfect knowledge of the domain, although now some of that knowledge may be indexical. It does not handle the basic indexical "I". It also does not allow indexical knowledge to be related to objective knowledge (e.g., after looking at a clock, agent knows what time it is).

Another conception of how reactive behavior can be produced has been proposed by Rosenschein and Kaelbling (1986) (Kaelbling and Rosenschein 1990). Their approach is of particular interest to us because a logic is used to specify reactive agents and the environments in which they operate. Design tools based on the logic are also provided; these tools facilitate the specification of complex agents and allow high-level specifications to be "compiled" into circuit-level ones.

The architecture they propose for reactive agents also avoids formal manipulation of explicit representations due to its inefficiency. But their agents are not purely functional machines; they do have state (registers). They also argue that the lack of explicit representations does not mean that one loses the ability to ascribe semantic content to machine states. They propose an alternative way of doing this — the situated automata view. It involves viewing the agent and its environment as coupled automata. A machine state contains the information that φ (i.e., in that state, the machine knows that φ) if given the state transition function of the machine and world, the fact that the machine is in that state implies that φ must be the case. For example, the fact that a certain wire in a robot carries a "1" would mean that there is an object within a certain radius of his position, if given the state transition function, the wire can only be carrying a "1" if there is an object within that radius. Thus the information content of a machine state is defined in terms of how it is correlated to environmental conditions.

Rosenschein and Kaelbling do not discuss the role of indexical information in the production of reactive behavior, nor how their framework handles it. It appears to accommodate temporally indexical knowledge but not temporally objective knowledge (much like the logic used by Moore (1980; 1985)). Indexicality with respect to the agent is not handled. It seems that a theory of knowledge and action that handles the distinction between indexical and objective knowledge might be very useful for producing better accounts of reactive behavior and better tools for the design of reactive agents.

Chapter 3

A Logic of Knowledge, Time, Action, and Ability

In this chapter, we propose a logical system that formalizes notions of knowledge, time, action, and ability. The epistemic and temporal parts of the logic adequately support the representation of indexical knowledge, keeping it distinct from *de re* knowledge. The complete system permits an adequate characterization of the indexical knowledge prerequisites and effects of actions in a wide range of circumstances. The first section gives an introduction to the main elements of the formalization and their semantics. The following two sections are precise specifications of the syntax and semantics of the system. Section 3.4 presents definitions for a large set of derived operators, as well as complex action expressions. Section 3.5 goes through all the notions formalized in the logic and discusses their properties. Section 3.6 proposes a formalization of ability as a derived notion. The last section discusses the main contributions and limitations of the system.

3.1 Introduction

Our theory of knowledge, time, action, and ability is embodied in a first-order modal logic.[1] The primary concern of our theory is the formalization of indexical knowledge, so let's start there. When one has indexical knowledge, for example, when one knows that one is currently

[1]An abridged description of the logic was published in (Lespérance and Levesque 1990). An early version was published in (Lespérance 1989). For general references to modal logic, see (Hughes and Cresswell 1968) or (Chellas 1980).

hungry, what is known is a "proposition" that is relative. It may be relative to the knower, or to the time of the knowing, or perhaps to other aspects of the context. One typically expresses such relative propositions with sentences containing context-sensitive elements such as 'I', 'now', 'here', 'this person', etc. Our logic reflects this: its formulas contain elements whose interpretation depends on the context as well as the objective circumstances. The logical symbols **self** and **now** are indexical terms that refer respectively to the agent component and time component of the context.[2] Non-logical symbols will also typically depend on the time of the context for their evaluation; for example, HUNGRY(a) might represent the fact that the agent assigned to variable a is hungry at the time of the context. They can also depend on the agent component of the context; for example, THIRSTY might mean that the agent of the context is thirsty at the time of the context.[3]

We model a context simply as a pair consisting of an agent and a time. This provides adequate semantics for many indexical expressions. For instance, the indexical **here** can be taken to stand for the term POS(**self**), that is, 'the position of the agent of the context' — we will in fact use this definition throughout this thesis.[4] In other cases, one may be forced to be more specific than would be required in a natural language; for instance, one may have to say something like 'the person at such and such relative position from **self** at time **now**' rather than 'this person'. Informally, our logic only captures indexical expressions that are functions of **self** and **now**.[5] We view the question of whether the semantics of indexicals like 'you' and demonstratives like 'this person' can be captured by treating them as functions of **self** and **now** as open. Such a reduction would undoubtedly be complex, but it is not clear how such indexical expressions could play a causal role in cognition without there being such a reduction. But since we are primarily concerned with modeling action and its relationship to knowledge rather with providing a formal semantics for natural language, we can afford to remain uncommitted on this issue.[6]

[2]Note that self and **now** are not intended to be formal counterparts of particular natural language words and often behave quite differently from any such counterparts.

[3]The idea of having formulas depend on the agent as well as the time of the context came from Grove and Halpern (1989).

[4]Such definitions also have the advantage of capturing the relationships between the indexical terms involved.

[5]Other choices of primitives are also possible. One might take **here** instead of self as a primitive and define the latter as 'the agent at position **here**'. Or one might even have a single primitive 'self-now' that refers to the current agent-state. We prefer the proposed approach because it seems most natural to take knowing or having a belief to be properties of an agent and a time.

[6]Lewis (1979) also appears to hold the view that the only contextual primitives needed are an agent and a time.

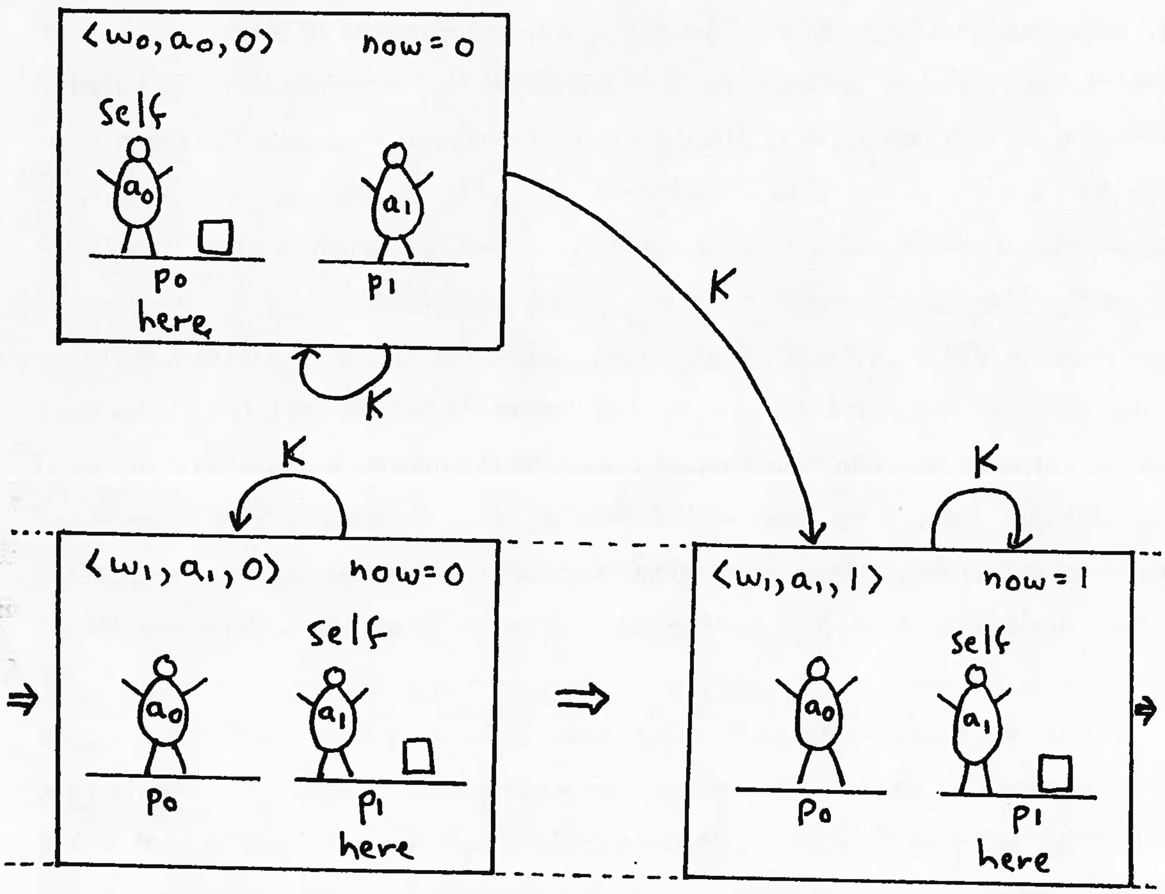
Thus, our semantics evaluates formulas with respect to *indices*, which consist of a possible world, modeling the objective circumstances, and an agent and time, modeling the context. We talk about a formula being satisfied by a model, index, and variable assignment.

Our semantics for knowledge is a simple generalization of the standard possible-world scheme (Kripke 1963; Hintikka 1969): the knowledge accessibility relation K is taken to hold over indices. $\langle\langle \mathsf{w},\mathsf{a},\mathsf{t}\rangle,\langle \mathsf{w}',\mathsf{a}',\mathsf{t}'\rangle\rangle \in \mathsf{K}$ iff as far as agent a at time t in world w knows, it may be the case that w' is the way the world actually is *and* he is a' *and* the current time is t'. Thus, we model the knowledge state of an agent at a time in a world by a set of indices, which characterizes not only which worlds are compatible with what the agent knows but also which points of view upon these worlds are compatible with what he knows.

The formula $\mathbf{Know}(\varphi)$ is used to represent the fact that at time **now** (i.e., the time of the context), **self** (i.e., the agent of the context) knows that φ. If φ contains indexical elements, $\mathbf{Know}(\varphi)$ should be taken as attributing indexical knowledge, that is, knowledge the agent has about himself and the current time. For example, if we use $\textsc{Holding}(x)$ to represent the fact that **self** is holding the object denoted by x at time **now**, then $\mathbf{Know}(\textsc{Holding}(x))$ would mean that **self** knows that *he himself* is currently holding the object denoted by x.

To get a better feel for how the account works, let's look at a few examples. The relevant parts of a semantic structure that models the knowledge state of an agent a_0 at a time 0 in a world w_0 are represented in figure 3.1. The domain of possible worlds $\mathcal{W}$, that of agents $\mathcal{A}$, and that of times $\mathcal{T}$, are specified at the top of the figure. As represented, there are two indices that are compatible with what a_0 knows at time 0 in w_0 (i.e., K-accessible): $\langle \mathsf{w}_0,\mathsf{a}_0,0\rangle$ itself and $\langle \mathsf{w}_1,\mathsf{a}_1,1\rangle$. This means that as far as a_0 knows in w_0 at time 0, it may be the case that the actual world is w_0, that he is a_0, and that the current time is time 0, or it may be the case that the actual world is w_1, that he is a_1, and that the current time is time 1; he cannot rule out either of these possibilities. Thus, we have that in world w_0 at time 0, agent a_0 does not know who he is — as far as he is concerned, he might be a_0 or he might be a_1. This is formally represented by statement (1a) at the bottom of the figure; it says that at index $\langle \mathsf{w}_0,\mathsf{a}_0,0\rangle$, it is not the case that there is someone whom the agent knows to be himself. Similarly, in world w_0 at time 0, agent a_0 does not know what time it is — as far as he is concerned, it might currently be time 0 or it might be time 1; this is what statement (1b) says. Also, in the two K-accessible indices, the denotation of **here** is

$\mathcal{W} = \{w_0, w_1\} \quad \mathcal{A} = \{a_0, a_1\} \quad \mathcal{T} = \mathbb{Z}$



$\langle w_0, a_0, 0\rangle \models \neg\exists a\ \mathrm{Know}(\mathbf{self} = a)$ (1a)
$\langle w_0, a_0, 0\rangle \models \neg\exists t\ \mathrm{Know}(\mathbf{now} = t)$ (1b)
$\langle w_0, a_0, 0\rangle \models \neg\exists p\ \mathrm{Know}(\mathbf{here} = p)$ (1c)
$\langle w_0, a_0, 0\rangle \models \mathrm{Know}(\exists b(\mathrm{BLOCK}(b) \wedge \mathrm{POS}(b) = \mathbf{here}))$ (1d)
$\langle w_0, a_0, 0\rangle \models \neg\exists a \exists t(a = \mathbf{self} \wedge t = \mathbf{now} \wedge$
$\mathrm{Know}(\mathrm{At}(t, \exists b(\mathrm{BLOCK}(b) \wedge \mathrm{POS}(b) = \mathrm{POS}(a)))))$ (1e)

Figure 3.1: A semantic structure in which an agent has indexical knowledge without having the corresponding *de re* knowledge

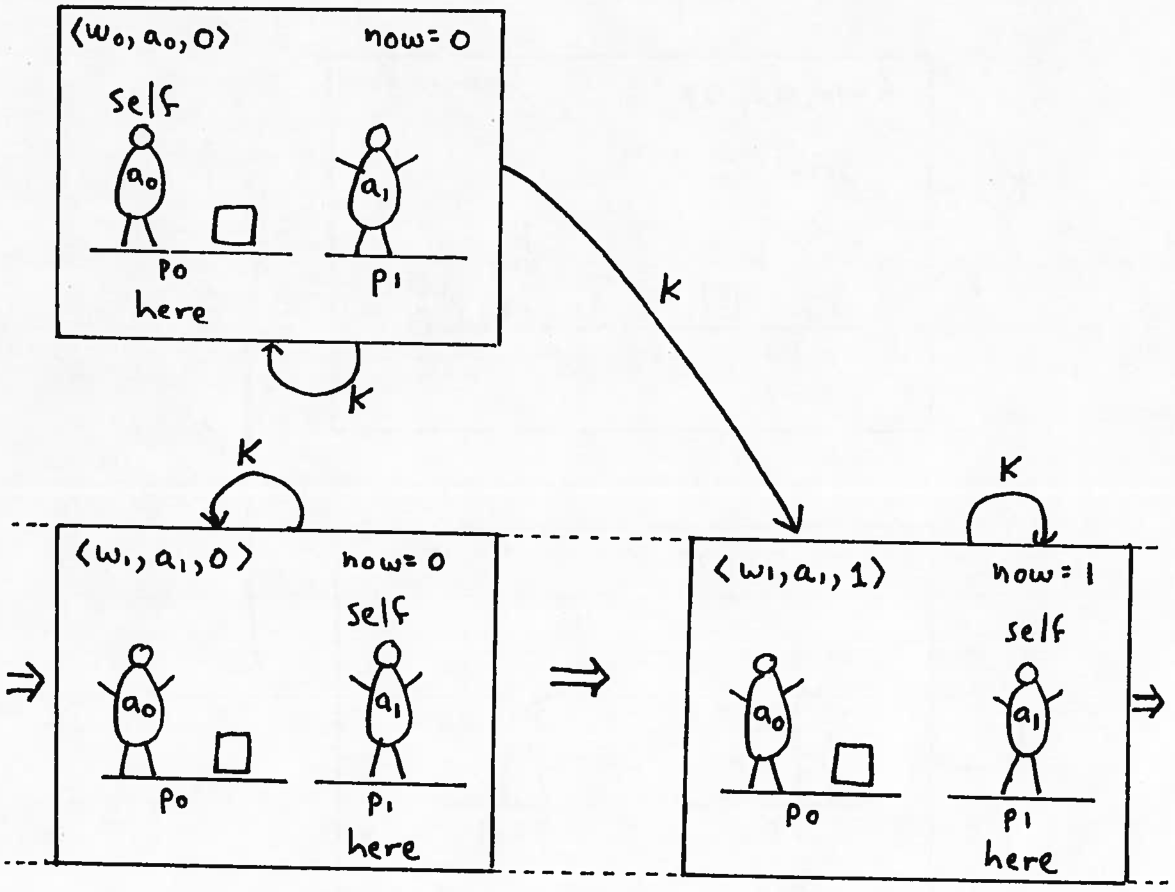
different; in one case it is p_0, while in the other, it is p_1. So the agent does not know where he is — statement (1c).

In spite of being ignorant about all this, the agent can have a lot of indexical knowledge, and our framework allows this to be modeled. For instance in the situation represented in figure 3.1, the agent knows that there is a block where he currently is — statement (1d). This is the case because $\exists b(\textsc{Block}(b) \wedge \textsc{pos}(b) = \mathbf{here})$ is true at all K-accessible indices. However, the agent does not have the *de re* knowledge that corresponds to this indexical knowledge; he does not know of himself and of the current time that there is a block where that agent is at that time — statement (1e). (The **At** operator serves to have its propositional argument evaluated at an index whose time component is t; this is explained below.) Note that (1e) is true at index $\langle w_0, a_0, 0\rangle$ because index $\langle w_1, a_1, 1\rangle$ is K-accessible from $\langle w_0, a_0, 0\rangle$, and at $\langle w_1, a_1, 1\rangle$, it is not the case that there is a block where a_0 is at time t_0, that is, at the earlier index $\langle w_1, a_1, 0\rangle$ (the box on the bottom left), there is no block where a_0 currently is. (The $\Rightarrow$ represents time flow; the indices $\langle w_1, a_1, 0\rangle$ and $\langle w_1, a_1, 1\rangle$ model different states of the same world history w_1, from the point of view of agent a_1.) This situation (the agent having indexical knowledge (1d) without having the corresponding *de re* knowledge (1e)) arises from the fact that he does not know who he is or what time it is.

Our framework also handles cases where the agent has some *de re* knowledge of himself and of the current time without having the corresponding indexical knowledge. For example, consider the variant of the previous situation modeled in figure 3.2. Agent a_0 at time 0 in world w_0 knows of himself and of the current time that there is a block where that agent is at that time — statement (2e). But a_0 in w_0 at time 0 does not know that there is a block where he himself currently is — statement (2d). This can occur because as in the previous situation, the agent fails to know who he is and what time it is.

Of course, it is also possible for an agent to know who he is and what time it is, and thus to have both indexical knowledge and the corresponding *de re* knowledge; this case is also handled by the framework. In the variant of the situation above represented in figure 3.3, agent a_0 at time 0 in world w_0 knows who he is, knows what time it is, and where he is (statements (3a), (3b), and (3c)). a_0 in w_0 at time 0 does know that there is a block where he himself currently is — statement (3d). And so a_0 in w_0 at time 0 also knows of himself and of the current time that there is a block where that agent is at that time — statement

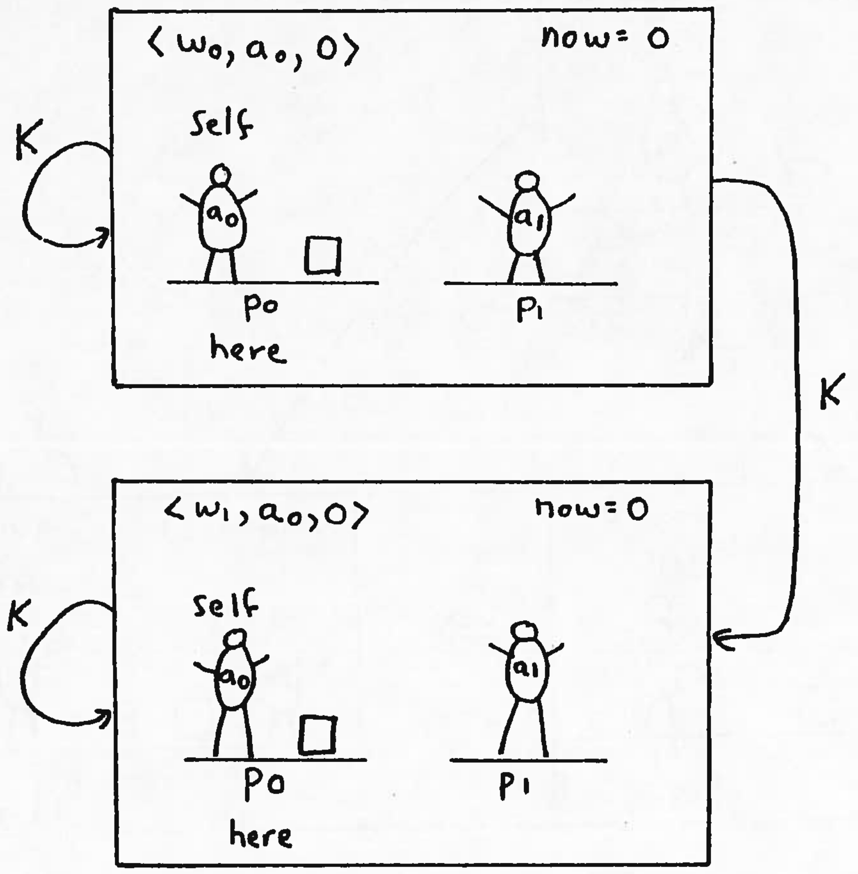
$\mathcal{W} = \{w_0, w_1\} \quad \mathcal{A} = \{a_0, a_1\} \quad \mathcal{T} = \mathbf{Z}$



$$\langle w_0, a_0, 0\rangle \models \neg\exists a\, \mathbf{Know}(\mathbf{self} = a) \tag{2a}$$
$$\langle w_0, a_0, 0\rangle \models \neg\exists t\, \mathbf{Know}(\mathbf{now} = t) \tag{2b}$$
$$\langle w_0, a_0, 0\rangle \models \neg\exists p\, \mathbf{Know}(\mathbf{here} = p) \tag{2c}$$
$$\langle w_0, a_0, 0\rangle \models \neg\mathbf{Know}(\exists b(\textsc{Block}(b) \wedge \textsc{pos}(b) = \mathbf{here})) \tag{2d}$$
$$\langle w_0, a_0, 0\rangle \models \exists a \exists t(a = \mathbf{self} \wedge t = \mathbf{now} \wedge \\ \mathbf{Know}(\mathbf{At}(t, \exists b(\textsc{Block}(b) \wedge \textsc{pos}(b) = \textsc{pos}(a))))) \tag{2e}$$

Figure 3.2: A structure in which an agent has *de re* knowledge without having the corresponding indexical knowledge

$\mathcal{W} = \{w_0, w_1\} \quad \mathcal{A} = \{a_0, a_1\} \quad \mathcal{T} = \mathbb{Z}$



$$\langle w_0, a_0, 0\rangle \models \exists a\, \mathbf{Know}(\mathbf{self} = a) \tag{3a}$$
$$\langle w_0, a_0, 0\rangle \models \exists t\, \mathbf{Know}(\mathbf{now} = t) \tag{3b}$$
$$\langle w_0, a_0, 0\rangle \models \exists p\, \mathbf{Know}(\mathbf{here} = p) \tag{3c}$$
$$\langle w_0, a_0, 0\rangle \models \mathbf{Know}(\exists b(\textsc{Block}(b) \wedge \textsc{pos}(b) = \mathbf{here})) \tag{3d}$$
$$\langle w_0, a_0, 0\rangle \models \exists a \exists t(a = \mathbf{self} \wedge t = \mathbf{now} \wedge \mathbf{Know}(\mathbf{At}(t, \exists b(\textsc{Block}(b) \wedge \textsc{pos}(b) = \textsc{pos}(a))))) \tag{3e}$$

Figure 3.3: A structure in which an agent has both indexical knowledge and the corresponding *de re* knowledge

(3e).

By the way, we can use these examples to clarify the claim, made in section 1.1, that the objective–indexical distinction is orthogonal to the *de re–de dicto* distinction. In the situation described in figure 3.1, agent a_0 at time t_0 in world w_0 knows that there is a block where he himself currently is — statement (1d); this is *de dicto* indexical knowledge. But he does not know of himself and of the current time that there is a block where that agent is at that time — statement (1e); he does not have objective *de re* knowledge of the fact that there is a block where a_0 is at t_0 in w_0. In the situation represented in figure 3.2, the reverse holds; the agent knows of himself and of the current time that there is a block where that agent is at that time — statement (2e) — but does not know that there is a block where he himself currently is — statement (2d); he has objective *de re* knowledge of the fact involved, but does not have *de dicto* indexical knowledge of it. In the situation described in figure 3.3, the agent knows this fact in both of the above ways and also knows it in an indexical *de re* way: he knows of himself and of the current time that that agent is himself and that that time is the current time and that there is a block where he currently is; formally, this yields:

$$\langle w_0, a_0, 0\rangle \models \exists a \exists t(\mathbf{Know}(a = \mathbf{self} \wedge t = \mathbf{now} \wedge \exists b(\textsc{Block}(b) \wedge \textsc{pos}(b) = \mathbf{here})))$$

Finally, a case where the agent would know the same fact in an objective *de dicto* way could be described by saying that the agent knows that there is a block where the shortest robot is at the moment that this robot turned two years old, when he is in fact the shortest robot and is just turning two years old; this can be represented formally as:

$$\begin{aligned}\langle w_0, a_0, 0\rangle \models &\mathbf{self} = \textsc{shortRob} \wedge \mathbf{now} = \textsc{secBirth}(\textsc{shortRob}) \wedge \\ &\quad \mathbf{Know}(\mathbf{At}(\textsc{secBirth}(\textsc{shortRob}), \exists b(\textsc{Block}(b) \wedge \textsc{pos}(b) = \textsc{pos}(\textsc{shortRob}))))\end{aligned}$$

Clearly, the agent need not have either the indexical knowledge corresponding to this or the *de re* knowledge corresponding to this. The *de re–de dicto* distinction has to do with whether the agent's knowledge is about a specific individual or about whoever satisfies a description. The objective–indexical distinction has to do with whether the knowledge refers to the agent himself or the current time, or whether it is about an objective fact.

An important advantage of our approach is that one can still model properties of knowledge by imposing constraints upon the accessibility relation K. We require K to be reflexive and transitive, which means that the principles of modal system S4 hold for knowledge. The approach can be adapted for modeling belief by changing the constraints. This treatment of indexical knowledge was inspired by informal proposals by Perry (1979) and especially Lewis (1979).

We want to be able to make both eternal and indexical temporal assertions, express relations between indexically specified and objectively specified times, talk about agents knowing what time it is, etc. Due to this, time is reified in our logic, that is, terms that denote time points are included. Ordinary predicates, for instance HOLDING in the example above, are taken to represent static relations. As we have seen, atomic formulas involving such predicates are taken to assert that the associated relation holds at time **now**. One asserts that an agent does an action using the logical symbol **Does**, which functions as a predicate. For example, **Does**(GRASP, t) can be used to represent the fact that **self** does the action of grasping, from **now** to time t. The operator **At** is used to say that a state of affairs holds at a time other than **now**. For example, **At**(t, HOLDING(x)) could mean that **self** is holding x at time t. The argument formula is evaluated at an index whose time is the denotation of t. We assume a linear ordering on times. Note that in combination with knowledge, a reified temporal logic such as ours is definitely more expressive than the alternatives; for instance, we can say that an agent knows that another agent knows what time it is without himself knowing it. The temporal aspects of the formalism were influenced by the work of Shoham (1987), Allen (1984), and McDermott (1982), as well as by the first-order temporal logic **R** described by Rescher and Urquhart (1971).

It is possible to express the occurrence of many types of complex actions using the constructs introduced above. We have developed a set of definitions that make it easy to state the occurrence of sequentially composed or conditional actions.[7] The action expression $(\delta_1; \delta_2)$ represents the sequential composition of the actions δ_1 and δ_2. The expression **if**$(\varphi, \delta_1, \delta_2)$ represents the action that consists in doing action δ_1 if the condition φ holds, and action δ_2 otherwise. One asserts that **self** does the action represented by expression δ from **now** to time t using the form **Does**(δ, t).

[7]Note that the expressive power of our temporal logic is not limited to this class of actions; actions involving nondeterminism, concurrency, multiple agents, definite times, etc. can be represented. But our formalization of ability is limited to actions belonging to this class.

The operator **By** is used to change the agent component of the context just as **At** is used to change its time component; one can thus say that an indexical "proposition" holds for an agent other than that of the context currently in effect. For example, $\mathbf{By}(a, \mathbf{Know}(\varphi))$ would say that agent a knows that φ. The argument formula is evaluated at an index whose agent component is the denotation of a.

Any account of the ability of agents to achieve goals by doing actions must consider not only the actions that are actually done by the agents, but also those that are merely physically possible for them; it must look at possible courses of events that differ from the actual course of events in the way the future unfolds.[8] In the framework of dynamic logic (Goldblatt 1987), upon which Moore's account of action is based (Moore 1980), both the operators available and the semantics do not separate the modal (possible courses of events) aspect involved in this kind of consideration from the temporal aspect.[9] But this is inadequate for our purposes. We want to be able to say that a state of affairs holds at a given absolute time, not just that it holds whenever some sequence of actions terminates. We want a single absolute time frame to apply globally across all possible courses of events (this would reflect the common-sense view that time, like space, is a neutral medium where events happen, one whose topography is unaffected by these events). And we also want to be able to say that something has *actually* happened. Due to this, we do not follow the dynamic logic scheme; instead, we develop an approach based on work in the area of Ockhamist logic; see (Thomason 1984) for a review of this area of philosophical logic. Our approach is based on one of the simplest system of Ockhamist logic, one that Thomason calls $T \times W$ necessity. In this scheme, there is a unique global domain of times, which is linearly ordered, and states of affairs are taken to hold with respect to both a world and a time. To account for the fact that in a given world at a given time, there will usually be several possible future courses of events besides the actual one, there is an additional accessibility relation that links the worlds that correspond to these possible futures to the

[8]This should not be taken to reflect a metaphysical commitment to some kind of irreducible notion of free will. In fact, we do believe that the actions of an agent are fully determined by its physical state and natural laws. Perhaps one could explain the notion of an action being physically possible for an agent as that of an action that would be done by another agent that is in the same physical state as the original one except for his beliefs and goals.

[9]In dynamic logic, one uses the form $[\delta]\,\varphi$ to say that in all states where the action δ terminates (when started now), φ holds. And the semantics involves a set of possible states and a family of accessibility relations over these states, one for each primitive action; a state s_e is accessible from another state s_s by the relation associated with an action d if and only if d initiated in state s_s might terminate in s_e.

current world at the given time; in fact, we view this as a family of accessibility relations, one for each time point. And these relations are used to interpret an additional modal operator $\Box$, where $\Box\varphi$ means that φ is historically necessary at time **now**, that is, necessary given everything that has happened until **now**.[10] The dual operator $\Diamond$ for historical possibility is defined in the usual way. Using these operators one can state that an action is possible for an agent at a time, in the sense that all its physical prerequisites are satisfied,[11] or that some effects necessarily hold at the conclusion of an action.

Consider the following example: an agent has been cooking, he has a roast in the oven which is now ready; if he turns the oven off, then the roast will be done, if he fails to do so, then the roast will get burnt. It is possible for him to follow either course of action. Figure 3.4 shows how this can be handled in our framework. There are two possible worlds, w_0 and w_1, which are mutually accessible in the historical necessity sense at time 0 (i.e., formally, $\mathsf{w}_0 \approx_0 \mathsf{w}_1$), meaning that they are identical in every respect up to time 0 inclusively. In w_0, agent a_0 does the action of turning the oven off from time 0 to time 1, and so the roast is done in w_0 at time 1. In w_1 on the other hand, a_0 does not turn off the oven, so the roast is burnt in w_1 at time 1. w_0 and w_1 are of course not mutually accessible in the historical necessity sense at time 1, as they differ in what happens after time 0. Given such a semantic structure, we have that it is historically possible for agent a_0 at time 0 in world w_0 to do the action TURNOFF from then until time 1, that is, $\Diamond\mathbf{Does}(\textsc{turnOff}, 1)$ holds at index $\langle \mathsf{w}_0, \mathsf{a}_0, 0\rangle$ — statement (1). This is the case because w_0 is accessible to itself at time 0 and a_0 does TURNOFF in w_0 from time 0 to time 1. But we also have that it is historically possible for a_0 at time 0 in w_0 not to do action TURNOFF from then until time 1 — statement (2). This is because w_1 is accessible from w_0 at time 0 and a_0 does not do TURNOFF in w_1 from time 0 to time 1.

We can also talk about the effects of chosing either of these courses of action. At $\langle \mathsf{w}_0, \mathsf{a}_0, 0\rangle$, if the agent does TURNOFF from **now** until time 1, it necessarily follows that the roast is done at time 1 — statement (3). This is because in all worlds accessible from w_0 at time 0 where a_0 does TURNOFF from time 0 to time 1, that is w_0, ROASTDONE holds

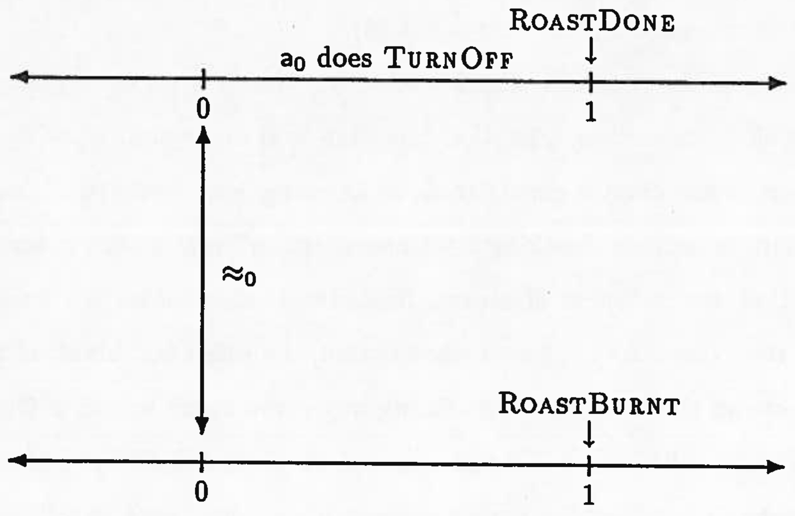
[10]We are probably not using the term 'historical necessity' in exactly the same sense as philosophers such as Thomason (1984): we take what is historically possible to be constrained by all sorts of assumptions regarding physical laws, properties of knowledge, etc.; thus, our notion of what is historically necessary goes well beyond the idea that it is impossible to change what has already happened.

[11]This should not be confused with ability, which requires the agent to *know of* the action that it is possible and results in the goal being achieved.

$$\mathcal{W} = \{w_0, w_1\} \quad \mathcal{T} = \mathbf{Z}$$

$$\approx_0 = \mathcal{W}^2 \quad \approx_1 = \{\langle w_0, w_0\rangle, \langle w_1, w_1\rangle\}$$

$$\begin{array}{ll} \langle w_0, a_0, 0\rangle \models \mathbf{Does}(\textsc{turnOff}, 1) & \langle w_0, a_0, 1\rangle \models \textsc{RoastDone} \\ \langle w_1, a_0, 0\rangle \models \neg\mathbf{Does}(\textsc{turnOff}, 1) & \langle w_1, a_0, 1\rangle \models \textsc{RoastBurnt} \end{array}$$



$$\begin{array}{lr}
\langle w_0, a_0, 0\rangle \models \Diamond\mathbf{Does}(\textsc{turnOff}, 1) & (1) \\
\langle w_0, a_0, 0\rangle \models \Diamond\neg\mathbf{Does}(\textsc{turnOff}, 1) & (2) \\
\langle w_0, a_0, 0\rangle \models \Box(\mathbf{Does}(\textsc{turnOff}, 1) \supset \mathbf{At}(1, \textsc{RoastDone})) & (3) \\
\langle w_0, a_0, 0\rangle \models \Box(\neg\mathbf{Does}(\textsc{turnOff}, 1) \supset \mathbf{At}(1, \textsc{RoastBurnt})) & (4) \\
\langle w_0, a_0, 1\rangle \models \Box(\mathbf{DoneFrom}(\textsc{turnOff}, 0) \wedge \textsc{RoastDone}) & (5) \\
\langle w_1, a_0, 1\rangle \models \Box(\neg\mathbf{DoneFrom}(\textsc{turnOff}, 0) \wedge \textsc{RoastBurnt}) & (6)
\end{array}$$

Figure 3.4: Modeling possible actions and their effects

at time 1. And we also have that at $\langle w_0, a_0, 0\rangle$, it is necessary that if the agent does not do TURNOFF from **now** until time 1, then the roast must be burnt at time 1 — statement (4). This is because in all worlds accessible from w_0 at time 0 where a_0 does not do TURNOFF from time 0 to time 1, that is w_1, ROASTBURNT holds at time 1.

Finally, notice that in world w_0 at time 1 it has become necessary that a_0 has done TURNOFF from time 0 to **now** and that the roast is done — statement (5). This is because these facts are no longer in the future and so must be necessary if they do hold. The semantic structure reflects this in that there are no worlds accessible from w_0 at time 1 where a_0 has not done TURNOFF from time 0 to **now** or the roast is not done. For similar reasons, in world w_1 at time 1 it is necessary that a_0 has not done TURNOFF from time 0 to **now** and that the roast is burnt — statement (6).

Finally, our theory includes a formalization of ability that is a revised version of that of Moore (1980). Recall from section 2.1.1 that one may well know that cooking beef bourguignon would impress the party's guests without knowing how to do it, that is, without knowing what primitive actions "cooking beef bourguignon" really stands for. This also applies to actions that are instances of general procedures: if one does not know the combination of a safe, then one does not know what dialing the safe's combination amounts to (even though one knows the procedure for dialing any given combination). One can view this distinction as an instance of *de dicto* as opposed to *de re* knowledge. Moore exploits this in his formalization of ability. For simple actions, his account goes as follows: an agent is able to achieve a goal by doing an action if and only if he knows what the given action is, and knows of himself that it is physically possible for him to do the action next and that his doing the action next necessarily results in the goal being achieved. By requiring that the agent know what the action is, Moore eliminates the need for explicit specification of the "knowledge prerequisites of actions": if an action is an instance of a general procedure and the procedure is known (formally, an epistemically rigid function), then the action is known iff the agent knows what its arguments stand for. Complex actions are handled recursively. Note that the agent is required to know who he is. Also note that it is not required that the agent initially know all the actions that make up a successful plan as long as he knows that he will know what to do next at each step of his plan.

Our formalization improves over Moore's in several ways. Firstly, we eliminate Moore's requirement that the agent know who he is; instead, we require indexical knowledge. Thus, in the simple action case we require the agent to know that it is physically possible for *himself* to do the action and that if *he himself* does the action, the goal will necessarily hold afterwards (as well as knowing what the action is). Mere *de re* knowledge is neither necessary nor sufficient for the agent to be able to achieve the goal (we give a concrete example of this in chapter 4). Secondly, it is based on a very expressive temporal logic. We can thus handle prerequisites or effects of actions that involve knowledge of absolute times and knowing what time it is, as well as several cases of actions that refer to times (e.g., locking up at 5 p.m.; see section 5.3). This would also make it easier to extend the formalization to handle more complex actions than are currently treated, for instance concurrent actions, actions involving several agents, and actions that refer to time in very general ways. Finally, the underlying logic includes an adequate treatment of indexical knowledge in general, which permits a more accurate specification of the knowledge prerequisites and effects of actions; the applications developed in chapters 4 and 5 are evidence for this. We use the formula $\mathbf{Can}(\delta, \varphi)$ to express the fact that **self** is able to achieve the goal φ by doing action δ.

Before moving on, we should discuss one last aspect of our framework. The formal notion of ability developed in this work is a highly simplified and idealized version of its common sense counterpart. It requires the agent to know enough to be absolutely guaranteed to achieve the goal, should he decide to do so and make effective use of his knowledge in the process. The real world is too complex and unpredictable to ever permit such a high degree of certitude and the common sense notion of ability that we use in deciding which intentions to form and which plans to adopt is more of the type "knowing a plan that leads to the goal being achieved with an appropriately high likelihood" rather than the "perfect" ability that we formalize.[12]

To ensure that agents do fulfill these requirements for ability in an interesting number of cases, we will need to make equally strong assumptions about the dynamics of agents' knowledge. Consider a typical situation where an agent must do an action involving several steps in order to achieve a goal. From his knowledge of the current state of affairs and of the first action, the agent comes to know that his doing this first step would result in some intermediate state of affairs holding. And from his knowledge of the remaining actions, it must be the case that he would be able to achieve his goal by doing these remaining steps,

[12]The same high degree of idealization is involved in our formalization of knowledge; this applies to nearly all current formalizations of the various propositional attitudes.

if he were to know that this intermediate state of affairs holds. Now if he is to be able to achieve the goal by doing the whole action, we must ensure that he would *realize* that the intermediate state of affairs holds after doing the first step. How do we do this? We could simply add the required assumptions in each individual case, but this would enormously complicate the modeling of such situations. What generalization would we be missing? Notice that if we assume that the agent must know what action he has just done and that he must remember what he knew, in this case, that his doing the first action would result in the intermediate state of affairs holding, then he must realize that this intermediate state of affairs holds. It is generally reasonable to assume that knowledge is persistent and that agents are aware of what primitive actions they have just done, and doing so removes the need for all the ad-hoc assumptions that must be made otherwise. So we will make this general assumption. Now in reality of course, agents not only happen to forget what they knew, but they may also have weak justifications for their knowledge or be convinced to renounce a true belief in spite of it being well justified. The assumption is another instance where we must idealize in order to get a workable formalization. Note that assuming persistence is much more reasonable for knowledge than for belief, since knowledge must be correct and (under the standard interpretation) justified; this is why our formalization involves knowledge rather than belief. Moore (1980) also makes this assumption. But the specification of the assumption is more complex in our framework because the temporal part of the logic is richer. Note also that indexical knowledge must be adjusted for the passage of time; for example, John's knowing that it is now raining must later become something like John's knowing that it was raining earlier. The exact form that the assumption takes and its main consequences are discussed in section 3.5.7.

3.2 Syntax

Our logic uses a many-sorted first-order modal language with equality $\mathcal{L}_{index}$. Table 3.1 includes a list of the syntactic categories involved, followed by a BNF-style specification of the non-primitive categories. The list of syntactic categories associates the name of each category (e.g., "Agent variable") with a symbol that is used to denote a typical member of the category (e.g., v^a). For primitive categories, the membership is also given. Different occurrences of the same symbol in a BNF rule are subscripted to allow reference to the

Syntactic categories

Category name	Syntactic variable	Membership
Agent variables	v^a	$\{a_0, a_1, \ldots\}$
Individual variables	v^i	$\{i_0, i_1, \ldots\}$
Temporal variables	v^t	$\{t_0, t_1, \ldots\}$
Action variables	v^d	$\{d_0, d_1, \ldots\}$
Agent functions	f^a	$\{\mathrm{F}^a_0, \mathrm{F}^a_1, \ldots\}$
Individual functions	f^i	$\{\mathrm{F}^i_0, \mathrm{F}^i_1, \ldots\}$
Temporal functions	f^t	$\{\mathrm{F}^t_0, \mathrm{T}^i_1, \ldots\}$
Action functions	f^d	$\{\mathrm{F}^d_0, \mathrm{F}^d_1, \ldots\}$
Predicate symbols	R	$\{\mathrm{R}_0, \mathrm{R}_1, \ldots\}$
Variables	v	
Agent terms	θ^a	
Individual terms	θ^i	
Temporal terms	θ^t	
Action terms	θ^d	
Formulas	φ	

Formation rules

$$
\begin{array}{rcl}
v & ::= & v^a \mid v^i \mid v^t \mid v^d \\
\theta^a & ::= & v^a \mid \mathbf{self} \mid f^a(\theta^i_1, \ldots, \theta^i_n) \\
\theta^i & ::= & \theta^a \mid v^i \mid f^i(\theta^i_1, \ldots, \theta^i_n) \\
\theta^t & ::= & v^t \mid \mathbf{now} \mid f^t(\theta^{i|t}_1, \ldots, \theta^{i|t}_n) \\
\theta^d & ::= & v^d \mid f^d(\theta^i_1, \ldots, \theta^i_n) \\
\varphi & ::= & R(\theta^i_1, \ldots, \theta^i_n) \mid \mathbf{Does}(\theta^d, \theta^t) \mid (\theta^a_1 = \theta^a_2) \mid (\theta^i_1 = \theta^i_2) \mid (\theta^t_1 = \theta^t_2) \mid (\theta^d_1 = \theta^d_2) \mid (\theta^t_1 < \theta^t_2) \mid \\
 & & \neg\varphi \mid (\varphi_1 \supset \varphi_2) \mid \forall v \varphi \mid \mathbf{At}(\theta^t, \varphi) \mid \mathbf{By}(\theta^a, \varphi) \mid \mathbf{Know}(\varphi) \mid \Box\varphi
\end{array}
$$

Table 3.1: Formal syntax of the logic.

expression denoted.

The syntax specified here is for the pure version of the logic. When the logic is used to formalize a particular domain, a theory is built on top of the logic — a theory that includes the validities of the logic as a subset of its own. In this case, a set of predicate and function symbols appropriate for modeling the domain is decided upon and the language of the theory is generated according to the rules as usual. This is the same distinction as that between the language of the predicate calculus and that of a particular first-order theory.[13]

We now go over the various forms in the language, giving for each an informal sketch of the intended meaning; the precise semantics is given in the next section. There are four sorts of terms: individual terms, agent terms, which are a subset of individual terms, action terms, which denote simple actions, and temporal terms, which denote time points. Compound terms of all four sorts may be formed; the arguments must be individual terms except in the case of temporal function applications, where the arguments can also be temporal terms. Constants are 0-argument functions; in this case, the parentheses are dropped. Also included are two indexical terms, **self** and **now**. As explained in the previous section, **self** and **now** respectively stand for the agent and time component of the index. The notions of free variable, bound variable, and term that is free for a variable in a formula are assumed to have their usual definition (Mendelson 1979). We use $\varphi\{v \mapsto \theta\}$ to stand for the result of substituting θ for all free occurrences of v in φ, provided that v and θ belong to the same sort.

An atomic formula of the form $\boldsymbol{R}(\theta^i_1, \ldots, \theta^i_n)$ is intended to mean that individuals $\theta^i_1, \ldots, \theta^i_n$ (strictly speaking, we should say the individuals denoted by $\theta^i_1, \ldots, \theta^i_n$ and similarly hereafter) stand in static relation $\boldsymbol{R}$ at time **now**. As explained earlier, the denotation of predicate and function symbols usually depends on the time of the index and may also depend on its agent. Another kind of atomic formula, $\mathbf{Does}(\theta^d, \theta^t)$, means that **self** does action θ^d from **now** to time θ^t. Other atomic formulas are identity statements and assertions concerning temporal ordering.

Formulas may be composed using the connectives and quantifiers of standard first-order logic, as well as a set of modal operators. $\mathbf{At}(\theta^t, \varphi)$ means that φ holds at time θ^t.

[13]For readability, we often use variables without subscripts or with non-numerical subscripts, as well as variables other than a, d, and t to stand for individuals; we assume that some systematic mapping of such abbreviations to variables actually in the language is used.

By(θ^a, φ) means that φ holds from the point of view of agent θ^a (i.e., φ holds at the index that only differs from the current one in that its agent component is that denoted by θ^a). **Know**(φ) means that self knows at time **now** that φ; when φ contains indexical elements, the formula is meant to attribute indexical knowledge. $\Box\varphi$ means that φ is historically necessary at time **now**, that is, necessary given everything that has happened up to and at time **now**. This completes the list of the primitives of our logic. Several derived forms are defined in section 3.4.

3.3 Semantics

In this section, we formally specify the semantics of our logic. The rationale behind the specification should generally be clear from the discussion in section 3.1.

3.3.1 Model Theory

A semantic structure M is a tuple

$$\langle \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{D}, \mathcal{W}, \prec, \mathsf{K}, \approx, \Phi, \Delta \rangle$$

The first four components are non-empty domains for the appropriate sorts: $\mathcal{A}$ is the domain of agents, $\mathcal{O}$ is the domain of objects, $\mathcal{T}$ is the domain of times, and $\mathcal{D}$ is the domain of primitive actions. The domain of individuals $\mathcal{I}$ is defined as $\mathcal{A} \cup \mathcal{O}$. $\mathcal{W}$ is a set of temporally extended possible worlds. $\mathcal{E} = \mathcal{W} \times \mathcal{A} \times \mathcal{T}$ is the set of indices. We take $\mathsf{a}, \mathsf{o}, \mathsf{i}, \mathsf{t}, \mathsf{d}, \mathsf{w}$, and e (possibly subscripted, primed, etc.), as ranging over arbitrary elements of $\mathcal{A}, \mathcal{O}, \mathcal{I}, \mathcal{T}, \mathcal{D}, \mathcal{W}$, and $\mathcal{E}$ respectively. $\prec$ is a relation over $\mathcal{T}$ whose intended interpretation is "is earlier than". $\mathsf{K} \subseteq \mathcal{E}^2$ is the knowledge accessibility relation. The rationale behind this formulation was explained in the previous section. $\approx$ is a family of accessibility relations — one for each time point — that is used to interpret the historical necessity operator $\Box$. Intuitively, $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$ if and only if w and w^* differ only in what happens after time t.

The denotation of terms and satisfaction of formulas are defined relative to indices. Φ gives the extension of predicate and function symbols at an index. $\Delta \subseteq \mathcal{D} \times \mathcal{E} \times \mathcal{T}$ determines which actions are done by which agents in which worlds over which time intervals: $\langle \mathsf{d}, \langle \mathsf{w}, \mathsf{a}, \mathsf{t}_s \rangle, \mathsf{t}_e \rangle \in \Delta$ iff action d is done by agent a from time t_s to time t_e in world w.

3.3.2 Denotation and Satisfaction

An assignment is a function that maps variables into elements of the domain appropriate to them. $\mathsf{g}\{v \mapsto \mathsf{x}\}$ is the assignment that is identical to g except that it maps variable v into the entity x.

The *denotation* of a term θ in a structure M at an index $\mathsf{e} = \langle \mathsf{w}, \mathsf{a}, \mathsf{t} \rangle$ under an assignment g, written $[\![\theta]\!]^{\mathsf{M}}_{\mathsf{e,g}}$ is defined as follows (from now on, we omit the structure under consideration when it is clear from context):

$[\![v]\!]_{\mathsf{e,g}} = \mathsf{g}(v)$.

$[\![\mathbf{self}]\!]_{\mathsf{e,g}} = \mathsf{a}$

$[\![\mathbf{now}]\!]_{\mathsf{e,g}} = \mathsf{t}$

$[\![f(\theta_1, \ldots, \theta_n)]\!]_{\mathsf{e,g}} = \Phi(f, \mathsf{e})([\![\theta_1]\!]_{\mathsf{e,g}}, \ldots, [\![\theta_n]\!]_{\mathsf{e,g}})$

We can now define what it means for a formula φ to be *satisfied* by a structure M, an index $\mathsf{e} = \langle \mathsf{w}, \mathsf{a}, \mathsf{t} \rangle$, and an assignment g, which we write $\mathsf{M}, \mathsf{e}, \mathsf{g} \models \varphi$:

$\mathsf{e}, \mathsf{g} \models \boldsymbol{R}(\theta^i_1, \ldots, \theta^i_n)$ iff $\langle [\![\theta^i_1]\!]_{\mathsf{e,g}}, \ldots, [\![\theta^i_n]\!]_{\mathsf{e,g}} \rangle \in \Phi(\boldsymbol{R}, \mathsf{e})$

$\mathsf{e}, \mathsf{g} \models \mathbf{Does}(\theta^d, \theta^t)$ iff $\Delta([\![\theta^d]\!]_{\mathsf{e,g}}, \mathsf{e}, [\![\theta^t]\!]_{\mathsf{e,g}})$

$\mathsf{e}, \mathsf{g} \models \theta_1 = \theta_2$ iff $[\![\theta_1]\!]_{\mathsf{e,g}} = [\![\theta_2]\!]_{\mathsf{e,g}}$

$\mathsf{e}, \mathsf{g} \models \theta^t_1 < \theta^t_2$ iff $[\![\theta^t_1]\!]_{\mathsf{e,g}} \prec [\![\theta^t_2]\!]_{\mathsf{e,g}}$

$\mathsf{e}, \mathsf{g} \models \neg\varphi$ iff it is not the case that $\mathsf{e}, \mathsf{g} \models \varphi$

$\mathsf{e}, \mathsf{g} \models (\varphi_1 \supset \varphi_2)$ iff either it is not the case that $\mathsf{e}, \mathsf{g} \models \varphi_1$, or $\mathsf{e}, \mathsf{g} \models \varphi_2$

$\mathsf{e}, \mathsf{g} \models \forall v \varphi$ iff for every entity x in the domain appropriate to v, $\mathsf{e}, \mathsf{g}\{v \mapsto \mathsf{x}\} \models \varphi$

$\mathsf{e}, \mathsf{g} \models \mathbf{At}(\theta^t, \varphi)$ iff $\langle \mathsf{w}, \mathsf{a}, [\![\theta^t]\!]_{\mathsf{e,g}} \rangle, \mathsf{g} \models \varphi$

$\mathsf{e}, \mathsf{g} \models \mathbf{By}(\theta^a, \varphi)$ iff $\langle \mathsf{w}, [\![\theta^a]\!]_{\mathsf{e,g}}, \mathsf{t} \rangle, \mathsf{g} \models \varphi$

$\mathsf{e}, \mathsf{g} \models \mathbf{Know}(\varphi)$ iff for all e', such that $\langle \mathsf{e}, \mathsf{e}' \rangle \in \mathsf{K}$, $\mathsf{e}', \mathsf{g} \models \varphi$

$\mathsf{e}, \mathsf{g} \models \Box\varphi$ iff for all w^* such that $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$, $\langle \mathsf{w}^*, \mathsf{a}, \mathsf{t} \rangle, \mathsf{g} \models \varphi$

A formula φ is *satisfiable* if there exists a structure M, index $\mathbf{e}$, and assignment $\mathbf{g}$, such that $\mathsf{M}, \mathbf{e}, \mathbf{g} \models \varphi$. A formula φ is *valid* (written $\models \varphi$) if it is satisfied by all structures, indices, and assignments.

3.3.3 Semantic Constraints

To ensure that the semantics adequately models the notions that we are trying to capture (knowledge, historical necessity, etc.), we impose various constraints on semantic structures.

Our first constraint ensures that what is known is true and that agents correctly introspect about what they know:

Constraint 3.1 K must be reflexive and transitive

As a result of this, the **Know** operator conforms to the principles of modal system S4. This is discussed in detail in section 3.5.2.

Constraints 3.2 and 3.3 are concerned with the modeling of time. Firstly, we require that time be linear:

Constraint 3.2
$\prec$ must be a strict total order, that is, transitive, connected, and irreflexive.

Secondly, we require that the ending time of any action not be earlier than its starting time:

Constraint 3.3 If $\langle \mathsf{d}, \langle \mathsf{w}, \mathsf{a}, \mathsf{t}_s \rangle, \mathsf{t}_e \rangle \in \Delta$, then $\mathsf{t}_s \preceq \mathsf{t}_e$.

Note that $\mathsf{t}_1 \preceq \mathsf{t}_2$ stands for the fact that either $\mathsf{t}_1 \prec \mathsf{t}_2$ or $\mathsf{t}_1 = \mathsf{t}_2$.

The next four constraints are concerned with the modeling of historical necessity. They ensure that historical alternatives at a given time differ only in what happens after that time. Constraint 3.4 specifies the static behavior of the historical necessity operator:

Constraint 3.4 For all $\mathsf{t} \in \mathcal{T}, \approx_{\mathsf{t}}$ must be an equivalence relation.

Note that an equivalence relation may be characterized as one that is reflexive and euclidean. A relation R is euclidean if and only if $\mathsf{x}_1 \mathsf{R} \mathsf{x}_2$ and $\mathsf{x}_1 \mathsf{R} \mathsf{x}_3$ implies $\mathsf{x}_2 \mathsf{R} \mathsf{x}_3$. As a consequence of constraint 3.4, $\Box$ and $\Diamond$ obey the principles of modal system S5. This is discussed further in section 3.5.5.

The other three constraints specify how what is necessary can change over time. Constraint 3.5 says essentially that possibilities do not increase as time passes:

Constraint 3.5 If $w \approx_{t_2} w^*$ and $t_1 \preceq t_2$, then $w \approx_{t_1} w^*$.

Constraint 3.6 ensures that historical alternatives at a given time are identical in what facts hold and what is known up to that time:

Constraint 3.6

If $w \approx_{t_2} w^*$ and $t_1 \preceq t_2$, then

1. for any predicate $\boldsymbol{R}$, $\Phi(\boldsymbol{R}, \langle w^*, a, t_1\rangle) = \Phi(\boldsymbol{R}, \langle w, a, t_1\rangle)$,
2. for any function symbol $\boldsymbol{f}$, $\Phi(\boldsymbol{f}, \langle w^*, a, t_1\rangle)(i_1, \ldots, i_n) = \Phi(\boldsymbol{f}, \langle w, a, t_1\rangle)(i_1, \ldots, i_n)$,
3. $\langle\langle w^*, a, t_1\rangle, e\rangle \in K$ iff $\langle\langle w, a, t_1\rangle, e\rangle \in K$.

And constraint 3.7 deals with action. It says that if w and w* are historical alternatives at time t, then (1) they should be identical in what primitive actions have been completed up to t, and (2) they should be identical in what primitive actions are under way at t, even though these actions can end at different times in w and w*.

Constraint 3.7

1. If $w \approx_t w^*$ and $t_e \preceq t$, then $\Delta(d, \langle w^*, a, t_s\rangle, t_e)$ iff $\Delta(d, \langle w, a, t_s\rangle, t_e)$,
2. if $w \approx_t w^*$, $t_s \prec t$ and $t \prec t_e$, then
   (a) if $\Delta(d, \langle w, a, t_s\rangle, t_e)$, then there exists t'_e such that $t \prec t'_e$ and $\Delta(d, \langle w^*, a, t_s\rangle, t'_e)$,
   (b) if $\Delta(d, \langle w^*, a, t_s\rangle, t_e)$, then there exists t'_e such that $t \prec t'_e$ and $\Delta(d, \langle w, a, t_s\rangle, t'_e)$.

The effects of these constraints are discussed further in section 3.5.5.

Finally, to simplify reasoning about agents' ability to achieve goals by doing multi-step actions, we will assume that knowledge is persistent, that is, that agents keep on knowing what they knew previously, and that agents know what actions they have done. The following constraint enforces this:

Constraint 3.8

If $\langle\langle w, a, t\rangle, \langle w', a', t'\rangle\rangle \in K$ and $t_p \preceq t$, then there exists a time t'_p, where $t'_p \preceq t'$, such that $\langle\langle w, a, t_p\rangle, \langle w', a', t'_p\rangle\rangle \in K$ and if $\Delta(d, \langle w, a, t_p\rangle, t)$ then $\Delta(d, \langle w', a', t'_p\rangle, t')$.

A detailed explanation of how this works is given in section 3.5.7. Note this formulation does not require agents to know at what time they start or finish acting, or how much time the action takes.

3.4 Definitions

In this section, we extend our language by defining various derived operators in terms of the primitives of the logic. In addition to providing useful tools for the developments in the remainder of the thesis, these definitions serve to display the expressive range of our logic.

Firstly, the standard propositional connectives, the existential quantifier, and various relational operators have their standard definitions. The historical possibility operator $\Diamond$ is also defined in the usual way.

Definition 3.1

$\vee, \wedge, \equiv, \exists, \neq, >, \leq, \geq$ are defined in the usual way;

$\Diamond\varphi \stackrel{\text{def}}{=} \neg\Box\neg\varphi$.

Next, using the operator **By**, we define a more common version of **Know** that specifies which agent knows the given proposition. Similar definitions are assumed for other operators that implicitly refer to an agent, such as **Does**, and others that are introduced below (**Res**, **Can**, etc.).

Definition 3.2

$\mathbf{Know}(\theta^a, \varphi) \stackrel{\text{def}}{=} \mathbf{By}(\theta^a, \mathbf{Know}(\varphi))$ and similarly for **Does**, **Res**, **Can**, etc.

We have developed a set of definitions that make it easy to state the occurrence of a large class of complex actions. Let us first define a new syntactic category, that of *action expressions*. We use syntactic variable δ to represent members of this category. The category is defined by the following BNF rule:

$$\delta ::= \theta^d \mid \mathbf{skip} \mid (\delta_1; \delta_2) \mid \mathbf{if}(\varphi, \delta_1, \delta_2)$$

It includes action terms, which represent simple actions, **skip**, which represents the empty action and takes no time, $(\delta_1; \delta_2)$, which represents the sequential composition of the actions δ_1 and δ_2, and $\mathbf{if}(\varphi, \delta_1, \delta_2)$, which represents the action that consists in doing action δ_1 if the condition φ holds, and in doing action δ_2 otherwise. One asserts that **self** does the action represented by expression δ from **now** to time θ^t using the form $\mathbf{Does}(\delta, \theta^t)$, which is now extended inductively to handle expressions as follows:

Definition 3.3

$\mathbf{Does}(\mathbf{skip}, \theta^t) \overset{\text{def}}{=} (\theta^t = \mathbf{now})$

$\mathbf{Does}((\delta_1;\delta_2), \theta^t) \overset{\text{def}}{=} \exists v_i^t(\mathbf{Does}(\delta_1, v_i^t) \wedge \mathbf{At}(v_i^t, \mathbf{Does}(\delta_2, \theta^t)))$, provided that v_i^t does not occur anywhere in θ^t, δ_1, or δ_2

$\mathbf{Does}(\mathbf{if}(\varphi, \delta_1, \delta_2), \theta^t) \overset{\text{def}}{=} (\varphi \wedge \mathbf{Does}(\delta_1, \theta^t)) \vee (\neg\varphi \wedge \mathbf{Does}(\delta_2, \theta^t))$

One base case is that where the action expression is an action term, in which case $\mathbf{Does}(\theta^d, \theta^t)$ is a primitive of the language. The case where δ is the empty action is also non-inductive: $\mathbf{Does}(\mathbf{skip}, \theta^t)$ simply stands for the requirement that the starting time **now** be equal to the ending time θ^t. The other two cases work inductively. For sequentially composed actions, the definition says that $\mathbf{Does}((\delta_1;\delta_2), \theta^t)$ stands for the fact that **self** does δ_1 from **now** to some intermediate time v_i^t and also does δ_2 from v_i^t to θ^t.[14] The final case is that of conditional actions: $\mathbf{Does}(\mathbf{if}(\varphi, \delta_1, \delta_2), \theta^t)$ stands for the fact that either the condition φ holds and **self** does the action δ_1 from **now** to θ^t, or φ doesn't hold and he does δ_2 from **now** to θ^t.

We also introduce three other forms, which are defined in terms of the previous ones: δ^{k}, which represents the sequential composition of k instances of δ ($\mathsf{k} \in \mathbb{N}$), $\mathbf{ifThen}(\varphi, \delta)$, a special case of **if** where there is nothing to be done if the condition does not hold, and $\mathbf{while}_{\mathsf{k}}(\varphi, \delta)$, which represents the action of doing δ up to k times as long as φ remains true ($\mathsf{k} \in \mathbb{N}$) — a bounded form of "while loop". These forms are defined as follows:

Definition 3.4

$$\delta^{\mathsf{k}} \overset{\text{def}}{=} \begin{cases} \mathbf{skip} & \text{if } \mathsf{k} = 0 \\ (\delta; \delta^{\mathsf{k}-1}) & \text{if } \mathsf{k} > 0 \end{cases}$$

$\mathbf{ifThen}(\varphi, \delta) \overset{\text{def}}{=} \mathbf{if}(\varphi, \delta, \mathbf{skip})$

[14]The proviso ensures that no free variable in δ_1 or δ_2, or in θ^t accidentally gets bound by the quantifier introduced by the definition. Since tests are the only places where a temporal variable can occur in an action expression, it would in fact be sufficient to require that v_i^t not occur free in θ^t or in any test of a conditional in δ_1 or δ_2 (this also applies to the other definitions in this section that introduce quantifiers and place an action expression within their scope). Note that the requirement that v_i^t not be free in θ^t also ensures that v_i^t does not get accidentally bound in the expansion of $\mathbf{Does}(\delta_1, v_i^t)$; for example if $\delta_1 = (\delta_3; \delta_4)$, then $\mathbf{Does}(\delta_1, v_i^t) \overset{\text{def}}{=} \exists v_j^t(\mathbf{Does}(\delta_3, v_j^t) \wedge \mathbf{At}(v_j^t, \mathbf{Does}(\delta_4, v_i^t)))$, where v_j^t is a different variable from v_i^t (and does not occur free in a test of δ_3 or δ_4).

$$\mathbf{while}_k(\varphi, \delta) \overset{\text{def}}{=} \begin{cases} \mathbf{skip} & \text{if } k = 0 \\ \mathbf{ifThen}(\varphi, (\delta; \mathbf{while}_{k-1}(\varphi, \delta))) & \text{if } k > 0 \end{cases}$$

By the way, note that the expressive power of our temporal logic is not limited to the class of actions that can be represented by our action expressions; the occurrence of actions involving nondeterminism, concurrency, multiple agents, definite times, etc. can also be expressed. Our special interest in this class comes from the fact that our formalization of ability is limited to actions that belong to it. Note however that actions involving indefinite iteration (or unbounded recursion) cannot be expressed in the logic. Since one can generally come up with a bound on the number of times an actual agent will be willing or able to repeat an action, this limitation does not appear to raise serious problems in practice. Moreover as discussed in section 2.1.1, the formalization of ability to do actions involving indefinite iteration is problematic.[15]

The next definitions introduce a set of operators that provide more flexibility in the specification of the times at which actions start and end:

Definition 3.5

$\mathbf{DoesNext}(\delta) \overset{\text{def}}{=} \exists v^t\, \mathbf{Does}(\delta, v^t)$, provided that v^t does not occur anywhere in δ

$\mathbf{DoneFromTo}(\delta, \theta^t_s, \theta^t_e) \overset{\text{def}}{=} \exists v^t_e(v^t_e = \theta^t_e \wedge \mathbf{At}(\theta^t_s, \mathbf{Does}(\delta, v^t_e)))$, provided that v^t_e does not occur anywhere in θ^t_s, θ^t_e, or δ

$\mathbf{DoneFrom}(\delta, \theta^t) \overset{\text{def}}{=} \mathbf{DoneFromTo}(\delta, \theta^t, \mathbf{now})$

$\mathbf{Done}(\delta) \overset{\text{def}}{=} \exists v^t_s\, \mathbf{DoneFrom}(\delta, v^t_s)$, provided that v^t_s does not occur anywhere in δ

The formula $\mathbf{DoesNext}(\delta)$ means that **self** does action δ from **now** to some future time. $\mathbf{DoneFromTo}(\delta, \theta^t_s, \theta^t_e)$ means that **self** does δ from time θ^t_s to time θ^t_e. $\mathbf{DoneFrom}(\delta, \theta^t)$ means that **self** does δ from time θ^t to **now**. And finally, $\mathbf{Done}(\delta)$ means that **self** does δ from some past time to **now**.

To facilitate reasoning about ability, we define the following notions:

[15]Also, extending the logic to handle indefinite iteration would seem to lead to incompleteness, as it involves quantification over all finite sequences of the iterated action.

Definition 3.6

PhyPoss(δ) $\overset{\text{def}}{=}$ $\Diamond\exists v^t$ **Does**(δ, v^t), where v^t is a temporal variable that does not occur free in δ

AfterNec(δ, φ) $\overset{\text{def}}{=}$ $\Box\forall v^t$(**Does**(δ, v^t) $\supset$ **At**(v^t, φ)), where v^t is a temporal variable that does not occur free in δ and φ

Res(δ, φ) $\overset{\text{def}}{=}$ **PhyPoss**(δ) $\wedge$ **AfterNec**(δ, φ)

PhyPoss(δ) means that it is "physically possible" for **self** to do action δ next (even though he may not be able to it because he does not know what primitive actions δ stands for). We say that φ must hold after δ, formally **AfterNec**(δ, φ), if and only if it is necessary that if **self** does δ next, then φ holds afterwards. We say that δ results in φ, formally **Res**(δ, φ), if and only if δ is physically possible and φ must hold after δ.

Finally, a few additional operators:

Definition 3.7

Kwhether(φ) $\overset{\text{def}}{=}$ **Know**(φ) $\vee$ **Know**($\neg\varphi$)

DoneWhen(δ, φ) $\overset{\text{def}}{=}$ $\exists v_s^t$(**DoneFrom**(δ, v_s^t) $\wedge$ **At**(v_s^t, φ)), provided that v_s^t does not occur anywhere in φ or δ

AllPast(φ) $\overset{\text{def}}{=}$ $\forall v^t(v^t <$ **now** $\wedge$ **At**(v^t, φ)), where v^t does not occur free in φ

SomePast(φ) $\overset{\text{def}}{=}$ $\neg$**AllPast**($\neg\varphi$)

AllFut(φ) $\overset{\text{def}}{=}$ $\forall v^t$(**now** $< v^t \wedge$ **At**(v^t, φ)), where v^t does not occur free in φ

SomeFut(φ) $\overset{\text{def}}{=}$ $\neg$**AllFut**($\neg\varphi$)

AllPastN(φ) $\overset{\text{def}}{=}$ **AllPast**(φ) $\wedge$ φ, and similarly for **AllFutN**

SomePastN(φ) $\overset{\text{def}}{=}$ **SomePast**(φ) $\vee$ φ, and similarly for **SomeFutN**

Kwhether(φ) means that self knows whether φ holds, **DoneWhen**(δ, φ) means that self has just done δ and that φ was true when he started, **AllPast**(φ) means that φ was true at all past times, **SomePast**(φ) means that φ was true at some past time, **AllFut**(φ) means that φ will be true at all future times, and **SomeFut**(φ) means that φ will be true at some future time. The above temporal operators also have variants **AllPastN**, **SomePastN**, **AllFutN**, and **SomeFutN** that talk about the present as well as the past or future. **AllPastN**(φ) means that φ holds **now** and at all past times, **SomePastN**(φ) means that φ holds either **now** or at some past time, and similarly for **AllFutN** and **SomeFutN**.

The ability operator **Can** is also defined in terms of the primitives; the definition is given in section 3.6.

3.5 Properties of the Logic

In this section, we present a fairly comprehensive set of properties that are validated by our logic. This clarifies the assumptions that are implicit in our semantics. Many of the properties will be useful in deriving proofs of ability, when we start formalizing particular domains. This should be viewed as only a first step toward obtaining a satisfactory axiomatization (a complete one, if possible).

3.5.1 The Basis

In this section, we present a set of properties that characterize the basis of our logic, that is, the part concerned with first-order logic with equality. This set of properties is basically the axiomatization proposed by Mendelson (1979) for first-order logic with equality with one important change (affecting proposition 3.3). This change reflects the fact that substitution of identicals does not necessarily preserve truth-value in our logic; using Quine's terminology, our modal operators are referentially opaque (Quine 1971). For example, we have that

$$\models \mathbf{Know}(\textsc{pres}\mathrm{US} = \textsc{pres}\mathrm{US})$$
$$\not\models \textsc{bush} = \textsc{pres}\mathrm{US} \supset \mathbf{Know}(\textsc{bush} = \textsc{pres}\mathrm{US})$$

that is, the fact that everyone knows that the president of the U.S.A. is the president of the U.S.A., together with the fact that George Bush happens to be the president of the U.S.A. does not imply one must know that the president of the U.S.A. is George Bush. Similar examples can be constructed for the other modal operators.

To reflect this we strengthen the normal restriction on the "axiom" of specialization. The result is proposition 3.3, which says that if $\forall v\varphi$ is true, then $\varphi\{v \mapsto \theta\}$ is also true, provided that θ is free for v in φ, *no occurrence of a function symbol gets substituted into the scope of a modal operator, no occurrence of* **self** *gets substituted into the scope of* **Know** *or* **By**, *and no occurrence of* **now** *gets substituted into the scope of* **Know** *or* **At**. This is similar to the approach taken by Levesque (1984a).

Let us then list the properties that characterize our basis:

Proposition 3.1 $\models \varphi$, where φ is an instance of a propositional tautology [16]

Proof:
Clearly, $\models \varphi_1 \supset (\varphi_2 \supset \varphi_1)$, $\models (\varphi_1 \supset (\varphi_2 \supset \varphi_3)) \supset ((\varphi_1 \supset \varphi_2) \supset (\varphi_1 \supset \varphi_3))$, and $\models (\neg\varphi_2 \supset \neg\varphi_1) \supset ((\neg\varphi_2 \supset \varphi_1) \supset \varphi_2)$. It is well known that these validities combined with the rule in proposition 3.4 form a complete axiomatization for the propositional calculus (Mendelson 1979). So all propositional tautologies must be valid in our logic. ■

Proposition 3.2 $\models \forall v(\varphi_1 \supset \varphi_2) \supset (\varphi_1 \supset \forall v \varphi_2)$, where v is not free in φ_1

Proof:
Assume that the formula is not valid. Then there exists e and g in some structure, such that $\mathsf{e},\mathsf{g} \models \forall v(\varphi_1 \supset \varphi_2)$, $\mathsf{e},\mathsf{g} \models \varphi_1$, and yet $\mathsf{e},\mathsf{g} \not\models \forall v \varphi_2$. The latter means that there exists an x, such that $\mathsf{e},\mathsf{g_x} \not\models \varphi_2$, where $\mathsf{g_x} = \mathsf{g}\{v \mapsto \mathsf{x}\}$. Since v is not free in φ_1 (by the proviso) and $\mathsf{e},\mathsf{g} \models \varphi_1$, we must also have that $\mathsf{M},\mathsf{e},\mathsf{g_x} \models \varphi_1$.[17] Since $\mathsf{e},\mathsf{g} \models \forall v(\varphi_1 \supset \varphi_2)$, we must have that $\mathsf{e},\mathsf{g_x} \models \varphi_1 \supset \varphi_2$. Combining the last two results yields that $\mathsf{e},\mathsf{g_x} \models \varphi_2$, which contradicts an earlier consequence of our assumption that the formula is not valid. ■

The following two lemmas set the stage for the statement and proof of our version of the axiom of specialization, proposition 3.3.

Lemma 3.1 $[\![\theta\{v \mapsto \theta'\}]\!]_{\mathsf{e},\mathsf{g}} = [\![\theta]\!]_{\mathsf{e},\mathsf{g}\{v \mapsto [\![\theta']\!]_{\mathsf{e},\mathsf{g}}\}}$

Proof: By induction on the number of function symbols in θ.

Lemma 3.2

$\mathsf{e},\mathsf{g} \models \varphi\{v \mapsto \theta\}$ iff $\mathsf{e},\mathsf{g}\{v \mapsto [\![\theta]\!]_{\mathsf{e},\mathsf{g}}\} \models \varphi$, provided that θ is free for v in φ, no occurrence of a function symbol gets substituted into the scope of a **Know**, **At**, **By**, or $\Box$ operator, no occurrence of **self** gets substituted into the scope of **Know** or **By**, and no occurrence of **now** gets substituted into the scope of **Know** or **At**.

[16]This "axiom" might seem strange to those unfamiliar with the way modal logics are axiomatized; but this is perfectly legitimate since the set of instances of propositional tautologies is a recursive set.

[17]This assumes that if v is not free in a formula φ, then $\mathsf{e},\mathsf{g}\{v \mapsto \mathsf{x}\} \models \varphi$ iff $\mathsf{e},\mathsf{g} \models \varphi$. This obviously holds and can be proved by induction on the structure of φ.

Proof:
We prove this by induction on the number of connectives, quantifiers, and modal operators in φ.
1. If $\varphi = \boldsymbol{R}(\theta_1^i, \ldots, \theta_n^i)$. Then

$$
\begin{aligned}
&\mathrm{e,g} \models \boldsymbol{R}(\theta_1, \ldots, \theta_n)\{v \mapsto \theta\} \\
&\quad \text{iff} \;\; \langle [\![\theta_1\{v \mapsto \theta\}]\!]_{\mathrm{e,g}}, \ldots, [\![\theta_n\{v \mapsto \theta\}]\!]_{\mathrm{e,g}} \rangle \in \Phi(\boldsymbol{R}, \mathrm{e}) \;\; \text{by the definition of satisfaction} \\
&\quad \text{iff} \;\; \langle [\![\theta_1]\!]_{\mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\}}, \ldots, [\![\theta_n]\!]_{\mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\}} \rangle \in \Phi(\boldsymbol{R}, \mathrm{e}) \;\; \text{by lemma 3.1} \\
&\quad \text{iff} \;\; \mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\} \models \boldsymbol{R}(\theta_1, \ldots, \theta_n) \;\; \text{by the definition of satisfaction}
\end{aligned}
$$

2. If $\varphi = \mathbf{Does}(\theta^d, \theta^t)$. Then

$$
\begin{aligned}
&\mathrm{e,g} \models \mathbf{Does}(\theta^d, \theta^t)\{v \mapsto \theta\} \\
&\quad \text{iff} \;\; \Delta([\![\theta^d\{v \mapsto \theta\}]\!]_{\mathrm{e,g}}, \mathrm{e}, [\![\theta^t\{v \mapsto \theta\}]\!]_{\mathrm{e,g}}) \;\; \text{by the definition of satisfaction} \\
&\quad \text{iff} \;\; \Delta([\![\theta^d]\!]_{\mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\}}, \mathrm{e}, [\![\theta^t]\!]_{\mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\}}) \;\; \text{by lemma 3.1} \\
&\quad \text{iff} \;\; \mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\} \models \mathbf{Does}(\theta^d, \theta^t) \;\; \text{by the definition of satisfaction}
\end{aligned}
$$

3. If $\varphi = (\theta_1 = \theta_2)$. Then

$$
\begin{aligned}
&\mathrm{e,g} \models \theta_1 = \theta_2\{v \mapsto \theta\} \\
&\quad \text{iff} \;\; [\![\theta_1\{v \mapsto \theta\}]\!]_{\mathrm{e,g}} = [\![\theta_2\{v \mapsto \theta\}]\!]_{\mathrm{e,g}} \;\; \text{by the definition of satisfaction} \\
&\quad \text{iff} \;\; [\![\theta_1]\!]_{\mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\}} = [\![\theta_2]\!]_{\mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\}} \;\; \text{by lemma 3.1} \\
&\quad \text{iff} \;\; \mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\} \models \theta_1 = \theta_2 \;\; \text{by the definition of satisfaction}
\end{aligned}
$$

4. $\varphi = (\theta_1 < \theta_2)$. The proof is similar to case 3.

For the remaining cases, assume that the property holds for all subformulas.
5. If $\varphi = \neg\varphi'$. Then

$$
\begin{aligned}
&\mathrm{e,g} \models \neg\varphi'\{v \mapsto \theta\} \\
&\quad \text{iff} \;\; \mathrm{e,g} \not\models \varphi'\{v \mapsto \theta\} \;\; \text{by the definition of satisfaction} \\
&\quad \text{iff} \;\; \mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\} \not\models \varphi' \;\; \text{by the induction hypothesis} \\
&\quad \text{iff} \;\; \mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\} \models \neg\varphi' \;\; \text{by the definition of satisfaction}
\end{aligned}
$$

6. If $\varphi = \varphi_1 \supset \varphi_2$. The proof is similar to case 5.
7. In case $\varphi = \forall v'\varphi'$. If $v = v'$, then the property trivially holds because there are no free occurrences of v in φ. If $v \neq v'$, we have that

$$
\begin{aligned}
&\mathrm{e,g} \models (\forall v'\varphi')\{v \mapsto \theta\} \\
&\quad \text{iff} \;\; \text{for all x in the appropriate domain, } \mathrm{e,g}\{v' \mapsto \mathrm{x}\} \models \varphi'\{v \mapsto \theta\} \;\; \text{by the def. of satisfaction} \\
&\quad \text{iff} \;\; \text{for all x, } \mathrm{e,g}\{v' \mapsto \mathrm{x}\}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}\{v' \mapsto \mathrm{x}\}}\} \models \varphi' \;\; \text{by the induction hypothesis} \\
&\quad \text{iff} \;\; \text{for all x, } \mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\}\{v' \mapsto \mathrm{x}\} \models \varphi' \\
&\quad \phantom{\text{iff}} \;\; \text{since by the proviso, } \theta \text{ must be free for } v \text{ in } \varphi \text{, and so } [\![\theta]\!]_{\mathrm{e,g}\{v' \mapsto \mathrm{x}\}} = [\![\theta]\!]_{\mathrm{e,g}} \\
&\quad \text{iff} \;\; \mathrm{e,g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\} \models \forall v'\varphi' \;\; \text{by the definition of satisfaction}
\end{aligned}
$$

8. If $\varphi = \mathbf{Know}(\varphi')$. In this case the proviso means we only need to prove the case where the term substituted is a variable, that is, where $\theta = v'$.

$\mathrm{e}, \mathrm{g} \models \mathbf{Know}(\varphi')\{v \mapsto v'\}$
 iff for all e' such that $\langle \mathrm{e}, \mathrm{e}' \rangle \in \mathsf{K}$, $\mathrm{e}', \mathrm{g} \models \varphi'\{v \mapsto v'\}$ by the def. of satisfaction
 iff for all e' such that $\langle \mathrm{e}, \mathrm{e}' \rangle \in \mathsf{K}$, $\mathrm{e}', \mathrm{g}\{v \mapsto [\![v']\!]_{\mathrm{e}',\mathrm{g}}\} \models \varphi'$ by the induction hypothesis
 iff for all e' such that $\langle \mathrm{e}, \mathrm{e}' \rangle \in \mathsf{K}$, $\mathrm{e}', \mathrm{g}\{v \mapsto [\![v']\!]_{\mathrm{e},\mathrm{g}}\} \models \varphi'$ since $[\![v]\!]_{\mathrm{e},\mathrm{g}} = [\![v]\!]_{\mathrm{e}',\mathrm{g}}$
 iff $\mathrm{e}, \mathrm{g}\{v \mapsto [\![v']\!]_{\mathrm{e},\mathrm{g}}\} \models \mathbf{Know}(\varphi')$ by the definition of satisfaction

9. If $\varphi = \mathbf{At}(\theta^t, \varphi')$. Let $\mathrm{e} = \langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle$ and $\mathrm{g}' = \mathrm{g}\{v \mapsto [\![\theta]\!]_{\mathrm{e},\mathrm{g}}\}$. Then

$\mathrm{e}, \mathrm{g} \models \mathbf{At}(\theta^t, \varphi')\{v \mapsto \theta\}$
 iff $\langle \mathrm{w}, \mathrm{a}, [\![\theta^t\{v \mapsto \theta\}]\!]_{\mathrm{e},\mathrm{g}} \rangle, \mathrm{g} \models \varphi'\{v \mapsto \theta\}$ by the definition of satisfaction
 iff $\langle \mathrm{w}, \mathrm{a}, [\![\theta^t]\!]_{\mathrm{e},\mathrm{g}'} \rangle, \mathrm{g} \models \varphi'\{v \mapsto \theta\}$
 since by lemma 3.1, $[\![\theta^t\{v \mapsto \theta\}]\!]_{\mathrm{e},\mathrm{g}} = [\![\theta^t]\!]_{\mathrm{e},\mathrm{g}'}$
 iff $\langle \mathrm{w}, \mathrm{a}, [\![\theta^t]\!]_{\mathrm{e},\mathrm{g}'} \rangle, \mathrm{g}\{v \mapsto [\![\theta]\!]_{\langle \mathrm{w}, \mathrm{a}, [\![\theta^t]\!]_{\mathrm{e},\mathrm{g}'} \rangle, \mathrm{g}}\} \models \varphi'$ by the induction hypothesis
 iff $\langle \mathrm{w}, \mathrm{a}, [\![\theta^t]\!]_{\mathrm{e},\mathrm{g}'} \rangle, \mathrm{g}' \models \varphi'$
 since the proviso means that if $\varphi'\{v \mapsto \theta\} \neq \varphi'$, then θ can only be a variable or self, in which case $[\![\theta]\!]_{\langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}} = [\![\theta]\!]_{\langle \mathrm{w}, \mathrm{a}, \mathrm{t}' \rangle, \mathrm{g}}$
 iff $\mathrm{e}, \mathrm{g}' \models \mathbf{At}(\theta^t, \varphi')$ by the definition of satisfaction

10. If $\varphi = \mathbf{By}(\theta^a, \varphi')$. The proof is similar to case 9.
11. If $\varphi = \Box\varphi'$. Then

$\langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g} \models (\Box\varphi')\{v \mapsto \theta\}$
 iff for all w^* such that $\mathrm{w} \approx_{\mathrm{t}} \mathrm{w}^*$, $\langle \mathrm{w}^*, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g} \models \varphi'\{v \mapsto \theta\}$ by the definition of satisfaction
 iff for all w^* such that $\mathrm{w} \approx_{\mathrm{t}} \mathrm{w}^*$, $\langle \mathrm{w}^*, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}\{v \mapsto [\![\theta]\!]_{\langle \mathrm{w}^*, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}}\} \models \varphi'$ by the induction hyp.
 iff for all w^* such that $\mathrm{w} \approx_{\mathrm{t}} \mathrm{w}^*$, $\langle \mathrm{w}^*, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}\{v \mapsto [\![\theta]\!]_{\langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}}\} \models \varphi'$
 since the proviso implies that if $\varphi'\{v \mapsto \theta\} \neq \varphi'$, then θ can only be a variable, self or **now**, in which case $[\![\theta]\!]_{\langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}} = [\![\theta]\!]_{\langle \mathrm{w}^*, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}}$
 iff $\mathrm{e}, \mathrm{g}\{v \mapsto [\![\theta]\!]_{\mathrm{e},\mathrm{g}}\} \models \Box\varphi'$ by the definition of satisfaction

■

Proposition 3.3 (Specialization)

$\models \forall v \varphi \supset \varphi\{v \mapsto \theta\}$, provided that θ is free for v in φ, no occurrence of a function symbol gets substituted into the scope of a **Know**, **At**, **By**, or $\Box$ operator, no occurrence of **self** gets substituted into the scope of **Know** or **By**, and no occurrence of **now** gets substituted into the scope of **Know** or **At**.

Proof:
Assume that $\mathrm{e}, \mathrm{g} \models \forall v \varphi$. This means that for all x in the appropriate domain, $\mathrm{e}, \mathrm{g}\{v \mapsto$

x$\} \models \varphi$. So in particular, $\mathrm{e},\mathrm{g}\{v \mapsto [\![\theta]\!]_{\mathrm{e,g}}\} \models \varphi$. By lemma 3.2 therefore, $\mathrm{e},\mathrm{g} \models \varphi\{v \mapsto \theta\}$, provided that θ is free for v in φ, no occurrence of a function symbol gets substituted into the scope of a **Know**, **At**, **By**, or $\Box$ operator, no occurrence of **self** gets substituted into the scope of **Know** or **By**, and no occurrence of **now** gets substituted into the scope of **Know** or **At**. ■

Proposition 3.4 (Modus Ponens) If $\models \varphi_1$ and $\models \varphi_1 \supset \varphi_2$, then $\models \varphi_2$

Proposition 3.5 (Universal Generalization) If $\models \varphi$, then $\models \forall v \varphi$

Proposition 3.6 (Reflexivity of Equality) $\models v = v$

The proofs of propositions 3.4, 3.5, and 3.6 are trivial.

The next lemma and proposition show that the substitution of identicals still preserves truth-value when only variables are substituted.

Lemma 3.3

If $[\![v]\!]_{\mathrm{e,g}} = [\![v']\!]_{\mathrm{e,g}}$, then $[\![\theta]\!]_{\mathrm{e,g}} = [\![\theta']\!]_{\mathrm{e,g}}$, where θ' arises from replacing some, but not necessarily all, occurrences of v by v' in θ,

Proof: By induction on the number of function symbols in θ.

Proposition 3.7

$\models v = v' \supset (\varphi \supset \varphi')$, where φ' arises from replacing some, but not necessarily all, free occurrences of v by v' in φ, provided that v' is free for the occurrences of v that it replaces,

Proof: By induction on the number of connectives, quantifiers, and modal operators in φ.

Note that the fact that the substitution of identicals preserves truth-value when only variables are substituted, combined with the fact that $\forall x(\mathbf{Know}(x = x))$ is valid (proposition 3.18 below), implies that $\forall x \forall y(x = y \supset \mathbf{Know}(x = y))$ is also valid (proposition 3.19 below). We do not see this as really objectionable, since we take it to concern *de re* as opposed to *de dicto* modality. But note that for example, it means that if one knows of someone seen on a beach that he is a pillar of the community and also knows of someone seen in a dark alley that he is a spy, and if the person seen on the beach and that seen in the dark alley are actually the same, then one must also know of the person on the beach that he is a spy (Quine 1971). This suggests that if one wants to explain people's behavior in terms of what *de re* knowledge they have, then one should exercise caution in ascribing *de re* knowledge to them.

Finally, we prove an additional important and useful property — that the substitution of equivalents preserves equivalence:

Proposition 3.8 (Substitution of Equivalents)
If $\models \varphi_s \equiv \varphi'_s$, then $\models \varphi \equiv \varphi'$, where φ' arises from φ by replacing some occurrences of φ_s by φ'_s.

Proof: By induction on the number of connectives, quantifiers, and modal operators in φ.

3.5.2 Knowledge

In this section, we go over the basic properties of knowledge in the logic, in order to clarify the results of the formalization and emphasize its differences from standard modal logics of knowledge (Hintikka 1962; Halpern and Moses 1985). Additional properties relating to the dynamics of knowledge are discussed in section 3.5.7.

Let us first establish that the logic really differentiates between indexical knowledge and *de re* knowledge. The following proposition shows that the logic does not require agents to know who they are (1), nor does it require them to know what time it is (2); they may have such *de re* knowledge (the formulas are satisfiable), but they need not (the formulas are not valid).

Proposition 3.9

1. $\exists a\, \mathbf{Know}(a = \mathbf{self})$ is satisfiable, but not valid

2. $\exists t\, \mathbf{Know}(t = \mathbf{now})$ is satisfiable, but not valid

As a result, our logic captures the fact that one can have indexical knowledge without having the corresponding *de re* knowledge and vice-versa. As the following proposition shows, one can know that some state of affairs holds **now** without knowing of some time that the state of affairs holds at that time (1); for example, I might know that I have just arrived home without knowing at what time I arrived. Also conversely, one can know of the current time that some state of affairs holds at that time without knowing that the state of affairs currently holds (3); for example, I might know that I have a meeting that starts at noon without knowing that my meeting is starting now. The same applies to indexicality with respect to the agent as opposed to time: someone might know that he himself has some property without knowing of anyone that he has the property (2), and an agent might know of himself that he has a property without knowing that he himself has the property (4).

Proposition 3.10

1. $\not\models \mathbf{Know}(\varphi) \supset \exists t\, \mathbf{Know}(\mathbf{At}(t, \varphi))$

2. $\not\models \mathbf{Know}(\varphi) \supset \exists a\, \mathbf{Know}(\mathbf{By}(a, \varphi))$

3. $\not\models \exists t(t = \mathbf{now} \wedge \mathbf{Know}(\mathbf{At}(t, \varphi))) \supset \mathbf{Know}(\varphi)$

4. $\not\models \exists a(a = \mathbf{self} \wedge \mathbf{Know}(\mathbf{By}(a, \varphi))) \supset \mathbf{Know}(\varphi)$

The purely epistemic part of the logic verifies all the validities of the modal system S4+BF. Note however that these valid schemas do not have the quite the same interpretation as in standard epistemic logic. In our system, the knowledge of an agent at a time does not only consist of objective propositions, but also of "indexical propositions", propositions whose truth-value depends on the context. So $\mathbf{Know}(\varphi)$ is perhaps best read as saying that the agent now knows that he himself currently has property φ. As we will see, this sometimes has unexpected effects. Now, let's go over these schemas. The first says that if the agent knows that φ_1 implies φ_2 and knows that φ_1, then he must also know that φ_2:

Proposition 3.11 $\models \mathbf{Know}(\varphi_1 \supset \varphi_2) \supset (\mathbf{Know}(\varphi_1) \supset \mathbf{Know}(\varphi_2))$

In the epistemic logic literature, this is called the distribution axiom and in modal logic generally, it is called the K axiom. This reflects the fact that under our semantics, agents must know all logical consequences of what they know. The logic also validates the fact that what is known must be true:

Proposition 3.12 $\models \mathbf{Know}(\varphi) \supset \varphi$

This is called the knowledge axiom in epistemic logic and the T axiom in modal logic generally. It holds as a result of the fact that the accessibility relation K is required to be reflexive (constraint 3.1). Note in this case how indexicality affects the way the schema should be read. Strictly speaking, the above says that if the agent knows that he himself currently has property φ, then he actually has property φ now. In saying that the indexical proposition φ is true, we must ensure that the indexical elements it contains are evaluated in the right context, that is, the one that has the knower as its agent component and the time at which the knowing is taking place as its time component. Thus the following variants of the above schema are not valid:

Proposition 3.13

1. $\not\models \mathbf{Know}(\theta^a, \varphi) \supset \varphi$

2. $\not\models \mathbf{At}(\theta^t, \mathbf{Know}(\varphi)) \supset \varphi$

In 1, the consequent need not hold because **self** may denote a different agent from θ^a; for example, the fact that John knows that he is hungry does not imply that the current agent is hungry. Similarly in 2, the consequent need not hold because **now** may denote a different time from θ^t; for example if I knew that I was hungry at noon, it does not mean that I am hungry now.

The logic also takes the view that agents introspect correctly about what they know. This yields the following proposition, which says that if the agent knows something, he knows that he knows it:

Proposition 3.14 (Positive Introspection) $\models \mathbf{Know}(\varphi) \supset \mathbf{Know}(\mathbf{Know}(\varphi))$

This is called the positive introspection axiom in epistemic logic and the 4 axiom in general modal logic. It corresponds to our requirement that the accessibility relation K be transitive (constraint 3.1). Note again that our handling of indexical knowledge affects how the above statement should be read; it says that if the agent knows that φ, then he knows that he himself currently knows that φ. Since an agent may not know who he is or what time it is, the following variants of the above formula are not valid:

Proposition 3.15

1. $\not\models \mathbf{Know}(a, \varphi) \supset \mathbf{Know}(a, \mathbf{Know}(a, \varphi))$

2. $\not\models \mathbf{At}(t, \mathbf{Know}(\varphi)) \supset \mathbf{At}(t, \mathbf{Know}(\mathbf{At}(t, \mathbf{Know}(\varphi))))$

Proof:
We construct a counter model for the formulas:

$$\mathcal{W} = \{w_0, w_1\} \quad \mathcal{A} = \{a_0, a_1\} \quad \mathcal{T} = \{t_0, t_1\}$$
$$K = \{\langle e, e\rangle, \text{ for all } e \in \mathcal{E}\} \cup \{\langle\langle w_0, a_0, t_0\rangle, \langle w_1, a_1, t_1\rangle\rangle\}$$
$$\langle w_0, a_0, t_0\rangle, g \models R \quad \langle w_1, a_1, t_1\rangle, g \models R \quad \langle w_1, a_0, t_1\rangle, g \not\models R \quad \langle w_1, a_1, t_0\rangle, g \not\models R$$
$$g(a_o) = a_0 \quad g(t_o) = t_0$$

1. $\{e | \langle\langle w_0, a_0, t_0\rangle, e\rangle \in K\} = \{\langle w_0, a_0, t_0\rangle, \langle w_1, a_1, t_1\rangle\}$. R is satisfied by g and each of these indices, so we must have that $\langle w_0, a_0, t_0\rangle, g \models$ **Know**(R), and thus also that $\langle w_0, a_0, t_0\rangle, g \models$ **Know**(a_o, R). However, $\langle w_1, a_0, t_1\rangle, g \not\models$ R, so we have that $\langle w_1, a_1, t_1\rangle, g \not\models$ **Know**(a_o, R). Thus, $\langle w_0, a_0, t_0\rangle, g \not\models$ **Know**(**Know**(a_o, (R)), as well as, $\langle w_0, a_0, t_0\rangle, g \not\models$ **Know**(a_o, **Know**(a_o, R)). So 1 is not valid.
2. Similar to 1. ∎

We do not assume that agents correctly introspect about what they don't know; thus we have that $\not\models \neg$**Know**(φ) $\supset$ **Know**($\neg$**Know**(φ)).[18] Our semantics assumes that there is a unique global domain of quantification that applies for all indices; thus the same agents, objects, actions, and times are assumed to exist at all indices. Agents cannot be ignorant as to which entities exist. This yields the following proposition, which says essentially that if everything is known to be φ, then it is known that everything is φ:

Proposition 3.16 $\models \forall v$ **Know**(φ) $\supset$ **Know**($\forall v \varphi$)

This is called the Barcan formula in modal logic. Note that the converse must also hold ($\models$ **Know**($\forall v \varphi$) $\supset \forall v$ **Know**(φ)); it is a consequence of propositions 3.11, 3.17, and the basis.

Finally, we have that all the validities of the logic are known:

Proposition 3.17 (Knowledge Generalization) If $\models \varphi$, then $\models$ **Know**(φ)

This is called the knowledge generalization rule in epistemic logic and the necessitation rule in general modal logic.

Note that from propositions 3.17, 3.6, and 3.5, it is easy to show that agents must know of any entity that it is identical to itself:

Proposition 3.18 $\models \forall x$(**Know**($x = x$))

Combined with proposition 3.7, this leads immediately to the indiscernibility of equals in knowledge contexts, which was discussed in section 3.5.1:

[18]This becomes valid if one requires K to be euclidean, that is, iff for all e, e', and e'', if $\langle e, e'\rangle \in K$ and $\langle e, e''\rangle \in K$, then $\langle e', e''\rangle \in K$.

Proposition 3.19 $\models \forall x \forall y(x = y \supset \textbf{Know}(x = y))$

But note that this applies only to variables (i.e., *de re* knowledge), not to arbitrary terms; in general, knowledge contexts are still opaque.

Propositions 3.12 and 3.14 provide justification for a point made earlier about our approach: that it remains possible to model important properties of knowledge by imposing constraints upon the knowledge accessibility relation, this being one of the most attractive features of standard possible-world semantics. In fact, the exact same constraints that give rise to major properties of knowledge in standard possible-world semantics yield suitable indexical versions of these properties in our scheme. This suggests that our specification of semantic structures for indexical knowledge is the right one to adopt if one wants to preserve compatibility with standard possible-world semantics.

Let us also point out that the approach can be adapted for modeling belief by changing some of the constraints imposed upon the accessibility relation, just as in standard possible-world semantics. If instead of requiring that K be reflexive, we require that it be serial,[19] then we get that knowledge need no longer be true (proposition 3.12 no longer holds), but must be consistent ($\models \textbf{Know}(\varphi) \supset \neg\textbf{Know}(\neg\varphi)$).

3.5.3 The Temporal Elements

In this section, we state and prove a set of properties describing the behavior of the temporal operator **At**, indexical **now**, and temporal predicates < and **Does**. Semantic constraint 3.2 requires time to be linearly ordered. The following three propositions simply reflect this requirement:

Proposition 3.20 (Transitivity of <) $\models \forall v_1^t \forall v_2^t \forall v_3^t (v_1^t < v_2^t \wedge v_2^t < v_3^t \supset v_1^t < v_3^t)$

Proposition 3.21 (Connectedness of <)) $\models \forall v_1^t \forall v_2^t (v_1^t < v_2^t \vee v_1^t = v_2^t \vee v_1^t > v_2^t)$

Proposition 3.22 (Irreflexivity of <) $\models \forall v^t \neg(v^t < v^t)$

We next describe the behavior of the **At** operator and its interactions with the indexical **now**. Note that this part of the logic is based on the first-order temporal logic **R** described by Rescher and Urquhart (1971). The following three propositions describe how the **At**

[19] K is serial iff for all e, there exists an e′ such that ⟨e, e′⟩ ∈ K.

operator interacts with connectives and quantifiers. Essentially, the operator may be pushed in or pulled out of connectives and quantifiers without affecting the truth-value of the formula (a change of bound variable may be needed in the quantifier case). The first proposition may be read as saying that $\neg\varphi$ holds at time θ^t if and only if it is not the case that φ holds at θ^t.

Proposition 3.23 $\models \mathbf{At}(\theta^t, \neg\varphi) \equiv \neg\mathbf{At}(\theta^t, \varphi)$

The second proposition says essentially that $\varphi_1 \supset \varphi_2$ holds at time θ^t if and only if φ_1's holding at time θ^t implies that φ_2 also holds at time θ^t.

Proposition 3.24 $\models \mathbf{At}(\theta^t, \varphi_1 \supset \varphi_2) \equiv \mathbf{At}(\theta^t, \varphi_1) \supset \mathbf{At}(\theta^t, \varphi_2)$

The third proposition can be read as saying that at time θ^t everything is φ if and only if it is the case that for everything, φ holds at time θ^t.

Proposition 3.25 $\models \mathbf{At}(\theta^t, \forall v\varphi) \equiv \forall v\, \mathbf{At}(\theta^t, \varphi)$, provided that v does not occur in θ^t

Proof:

$\langle \mathrm{w}, \mathrm{a}, \mathrm{t}\rangle, \mathrm{g} \models \mathbf{At}(\theta^t, \forall v\varphi)$

iff $\langle \mathrm{w}, \mathrm{a}, [\![\theta^t]\!]_{\langle \mathrm{w},\mathrm{a},\mathrm{t}\rangle,\mathrm{g}}\rangle, \mathrm{g} \models \forall v\varphi$

iff for all x in the appropriate domain, $\langle \mathrm{w}, \mathrm{a}, [\![\theta^t]\!]_{\langle \mathrm{w},\mathrm{a},\mathrm{t}\rangle,\mathrm{g}}\rangle, \mathrm{g}\{v \mapsto \mathrm{x}\} \models \mathbf{At}(\theta^t, \varphi)$

iff for all x in the appropriate domain, $\langle \mathrm{w}, \mathrm{a}, [\![\theta^t]\!]_{\langle \mathrm{w},\mathrm{a},\mathrm{t}\rangle,\mathrm{g}\{v\mapsto \mathrm{x}\}}\rangle, \mathrm{g}\{v \mapsto \mathrm{x}\} \models \mathbf{At}(\theta^t, \varphi)$
since v does not occur in θ^t

iff for all x in the appropriate domain, $\langle \mathrm{w}, \mathrm{a}, \mathrm{t}\rangle, \mathrm{g}\{v \mapsto \mathrm{x}\} \models \mathbf{At}(\theta^t, \varphi)$

iff $\langle \mathrm{w}, \mathrm{a}, \mathrm{t}\rangle, \mathrm{g} \models \forall v\, \mathbf{At}(\theta^t, \varphi)$

■

The next two propositions deal with the interactions between the indexical **now** and the **At** operator, and the effects of iterating **At** operators. On the matter of **now**, notice that saying that something holds at time **now** is redundant because formulas are always evaluated with respect to **now**. So a state of affairs φ holds at time **now** if and only if it (simply) holds. This yields the following proposition:

Proposition 3.26 $\models \mathbf{At}(\mathbf{now}, \varphi) \equiv \varphi$

In effect, **now** behaves much like an identity element for **At**.

On the matter of iterated **At** modalities, notice that asserting that a state of affairs holds at a given time, where the time is objectively specified, for instance by means of a variable, results in a proposition that is temporally objective, one whose truth is independent of the time of evaluation; so nesting such a proposition in additional **At** operators has no effect. For example, if I say that it rained in Toronto on September 21, 1990, the resulting proposition is temporally objective; saying that on September 22, 1990, it was the case that on September 21, 1990 it rained in Toronto yields an equivalent proposition and is totally redundant. The following proposition expresses this; it says that φ's holding at time v^t holds at time θ^t if and only if φ holds at time v^t.

Proposition 3.27 $\models \mathbf{At}(\theta^t, \mathbf{At}(v^t, \varphi)) \equiv \mathbf{At}(v^t, \varphi)$

Note that the formula is no longer valid if v^t is replaced by a term that is not temporally rigid.

Finally, we have a generalization rule for **At**. As the following proposition says, if some proposition is valid, then it must hold at all times:

Proposition 3.28 (Temporal Generalization) If $\models \varphi$, then $\models \mathbf{At}(\theta^t, \varphi)$

There is only one logical restriction on the predicate **Does**: as required by semantic constraint 3.3, the ending time of an action cannot be earlier that its starting time. This yields the following proposition:

Proposition 3.29 $\models \mathbf{Does}(\theta^d, \theta^t) \supset \mathbf{now} \leq \theta^t$

3.5.4 The By Operator

The logical behavior of the modal operator **By** and the associated indexical **self** are in almost perfect correspondence with that of the **At** operator and the indexical **now**; all the logical principles respected by the latter pair are also obeyed by the former. Like **At**, the **By** operator may be pushed in or pulled out of connectives and quantifiers without affecting the truth-value of the formula (in the case of quantifiers, one may need to change bound variables). The next three propositions verify this. The first says that $\neg\varphi$ holds for agent θ^a if and only if it is not the case that φ holds for θ^a (perhaps a better reading is that θ^a has the property $\neg\varphi$ iff it is not the case that θ^a has property φ).

Proposition 3.30 $\models \mathbf{By}(\theta^a, \neg\varphi) \equiv \neg\mathbf{By}(\theta^a, \varphi)$

We omit the proofs for this and subsequent propositions, as they are exact analogues of those given for the corresponding principles for **At** in the previous section. The second proposition says that $\varphi_1 \supset \varphi_2$ holds for agent θ^a if and only if φ_1's holding for agent θ^a implies that φ_2 also holds for θ^a.

Proposition 3.31 $\models \mathbf{By}(\theta^a, \varphi_1 \supset \varphi_2) \equiv \mathbf{By}(\theta^a, \varphi_1) \supset \mathbf{By}(\theta^a, \varphi_2)$

The third proposition says that for agent θ^a everything is φ if and only if it is the case that for everything, φ holds for agent θ^a.

Proposition 3.32 $\models \mathbf{By}(\theta^a, \forall v \varphi) \equiv \forall v\, \mathbf{By}(\theta^a, \varphi)$, provided that v does not occur in θ^a

Now let's talk about how **self** and **By** interact and the effects of iterating **By** operators. Saying that something holds for agent **self** is just as redundant as saying that something holds at time **now**, since formulas are always evaluated with respect to **self**. As the following proposition says, φ holds for agent **self** if and only if it (simply) holds.

Proposition 3.33 $\models \mathbf{By}(\mathbf{self}, \varphi) \equiv \varphi$

Thus, **self** behaves like an identity element for **By**. As far as iterated **By** operators are concerned, the situation is similar to that with **At**: to say that something holds for an agent, where the agent is objectively specified, for instance by means of a variable, is to say something that no longer depends on the agent of the context; so nesting such an assertion in further **By** operators has no effects. Thus we get the following proposition, which says that φ's holding for agent v^a holds for agent θ^a if and only if φ holds for v^a.

Proposition 3.34 $\models \mathbf{By}(\theta^a, \mathbf{By}(v^a, \varphi)) \equiv \mathbf{By}(v^a, \varphi)$

The formula is no longer valid if v^a is replaced by a term that may be indexical with respect to the agent.

We also have a generalization rule for **By**, which says that if some proposition is valid, then it must hold for all agents:

Proposition 3.35 If $\models \varphi$, then $\models \mathbf{By}(\theta^a, \varphi)$

Finally, let's describe how the **By** operator interacts with the **At** operator. Essentially, the operators commute provided that the agent argument of **By** and the time argument of **At** contain no function symbols. Thus, we get the following proposition, which says that from the point of view of agent θ^a, φ holds at time θ^t if and only if at time θ^t, φ holds for agent θ^a, given that the proviso holds provided that θ^a and θ^t contain no function symbols (if θ^a or θ^t did contain a function symbol, then its denotation may be affected by the change in context).

Proposition 3.36
$\models \mathbf{By}(\theta^a, \mathbf{At}(\theta^t, \varphi)) \equiv \mathbf{At}(\theta^t, \mathbf{By}(\theta^a, \varphi))$, provided θ^a and θ^t contain no function symbols

3.5.5 Historical Necessity

In this section we describe the logical behavior of the historical necessity operator $\Box$ (and the historical possibility operator $\Diamond$ derived from it). As we have seen in section 3.1, the intuition behind historical necessity is that at any given point in time, what has happened until then is necessary (it's impossible to change it), while what will happen in the future still depends on what agents decide to do. The future is viewed as holding many possibilities. As time passes, some of these possibilities are actualized and thus become necessary, while others are not and thus become impossible.

From a logical standpoint, the behavior of the historical necessity operator is probably best understood by viewing it as an S5 modality that obeys some additional principles specific to historical necessity. Let us first describe the principles of the S5 modal system, as they apply to historical necessity. The first is a kind of distribution principle as we had for knowledge (in modal logic terminology, the K axiom). It says that if it is (historically) necessary that φ_1 implies φ_2, and φ_1 is necessary, then φ_2 must also be necessary.

Proposition 3.37 $\models \Box(\varphi_1 \supset \varphi_2) \supset (\Box\varphi_1 \supset \Box\varphi_2)$

The second principle says that if something is necessary, then it must be true.

Proposition 3.38 $\models \Box\varphi \supset \varphi$

This is called the T axiom in modal logic. Its validity results from our requirement that the historical necessity accessibility relation $\approx_{\mathrm{t}}$ be reflexive, in semantic constraint 3.4. The third principle says that if something is possible, then it is necessarily the case that it is possible.

Proposition 3.39 $\models \Diamond\varphi \supset \Box\Diamond\varphi$

This is called the "5" axiom in modal logic. It arises from the requirement that $\approx_{\mathrm{t}}$ be euclidean, as part of constraint 3.4. Finally, we have the following "necessitation" rule: if φ is valid, then $\Box\varphi$ must also be valid.

Proposition 3.40 (Necessitation) If $\models \varphi$, then $\models \Box\varphi$

Before moving on, let us mention a few consequences of the S5 principles together with our basis. Firstly, if φ is necessary, then it is necessarily the case that φ is necessary.

Proposition 3.41 $\models \Box\varphi \supset \Box\Box\varphi$

This is called the "4" axiom in modal logic and it is well known to be valid in S5. Another consequence of S5 and our basis is that everything is necessarily φ if and only if it is necessarily the case that everything is φ.

Proposition 3.42 $\models \forall v\Box\varphi \equiv \Box\forall v\varphi$

The left-to-right part of this double implication is the Barcan formula. Finally, another consequence is that if something is necessarily φ, then it is necessary that something is φ.

Proposition 3.43 $\models \exists v\Box\varphi \supset \Box\exists v\varphi$

Note that the converse of this formula is not valid; the fact that in all $\approx$-accessible worlds some entity is φ does not imply that there is an entity that is φ in all these accessible worlds. Hughes and Cresswell (1968) provide proofs of the above validities from a basis that is slightly different from ours; the proofs are easily adapted to our system.

Let's now talk about the logical properties of historical necessity that are specific to the notion, that is, that are not consequences of the basis and the S5 principles. The main idea behind these principles is roughly the following: any true statement that does not make claims about the future is historically necessary. Let us define the notion of a *future-blind* formula:

φ is *future-blind* if and only if φ contains no occurrences of the **At** operator or **Does** predicate outside the scope of a **Know** operator except in forms that can be represented as **SomePast** and **DoneWhen**.

Note that this notion does not cover all cases where a formula makes no claims about the future, but it does cover most cases of interest to us. The rest of this section will be mainly concerned with showing that a future-blind formula is historically necessary if and only if it is true (proposition 3.59).

To set the stage for this, we need to establish various results about how the various syntactic forms behave with respect to historical necessity. First, let's look at terms. In our logic, only the present attributes of an entity may play a role in determining whether it is denoted by a term; terms cannot refer to entities on the basis of their past or future attributes. This is reflected in the formal semantics, where function symbols are evaluated at time **now**. Thus, it would be inappropriate to try to represent a description that talks about future attributes such as "the president of the U.S.A. in year 2000" by a term in our language; rather, one should use an open formula $\varphi(a)$ that contains an **At** operator, such as $\exists t\ \mathbf{At}(t, \text{IsYear2000} \wedge a = \text{presUS})$. Since the present is necessary, this means that whatever a term denotes, it must necessarily denote it, that is formally, that the term must have the same denotation in all historical alternatives to the current world. This is exactly what the following proposition says:

Lemma 3.4 If $\mathrm{w} \approx_{\mathrm{t}} \mathrm{w}^*$, then $[\![\theta]\!]_{\langle \mathrm{w,a,t}\rangle,\mathrm{g}} = [\![\theta]\!]_{\langle \mathrm{w^*,a,t}\rangle,\mathrm{g}}$.

Proof: By induction on the structure of θ.

The situation is similar for atomic formulas, except for those of the form $\mathbf{Does}(\theta^d, \theta^t)$. An ordinary predicate application $R(\theta^i_1, \ldots, \theta^i_n)$ is intended to assert that the individuals denoted by the arguments stand in the static relation denoted by the predicate *at the current time*. This, together with what was just observed about the denotation conditions of terms, means that the truth of such an assertion cannot depend on future (or past) attributes of entities. Thus, if the state of affairs described by such an assertion holds, then it must necessarily hold, and if it doesn't hold, then it must be necessary that it doesn't hold. For assertions of identity and assertions about temporal ordering, the same principle applies, since these relations hold or fail to hold independently of time. However, the principle fails for assertions of the form $\mathbf{Does}(\theta^d, \theta^t)$, since such assertions state that the agent does action θ^d from **now** to some *future* time θ^t. The following two propositions verify this.

Proposition 3.44 $\models \Box\varphi \vee \Box\neg\varphi$, provided φ is atomic and not of the form $\mathbf{Does}(\theta^d, \varphi)$

Proposition 3.45 $\not\models \Box\mathbf{Does}(\theta^d, \theta^t) \vee \Box\neg\mathbf{Does}(\theta^d, \theta^t)$

Proof:
We construct a counter model. Let $\mathcal{W} = \{\mathrm{w_0, w_1}\}$, $\mathcal{T} = \{\mathrm{t_0, t_1}\}$, and $\mathrm{w_0} \approx_{\mathrm{t_0}} \mathrm{w_1}$. Let $\langle \mathrm{d}, \langle \mathrm{w_0, a, t_0}\rangle, \mathrm{t_1}\rangle \in \Delta$ and $\langle \mathrm{d}, \langle \mathrm{w_1, a, t_0}\rangle, \mathrm{t_1}\rangle \notin \Delta$. Finally, let $\mathrm{g}(d) = \mathrm{d}$ and $\mathrm{g}(t_1) = \mathrm{t_1}$. Then we have that $\langle \mathrm{w_0, a, t_0}\rangle, \mathrm{g} \models \mathbf{Does}(d, t_1)$ and yet $\langle \mathrm{w_0, a, t_0}\rangle, \mathrm{g} \models \Diamond\neg\mathbf{Does}(d, t_1)$. ■

Now, let us look at the interactions between historical necessity and the other modal operators: under what conditions is a modal formula necessary? Assertions of the form

Know(φ) state that the agent knows at time **now** that φ. Such formulas really talk only about the present knowledge state of the agent, even when the propositional argument φ contains references to the future. Thus, the same principle that we had for ordinary predicate applications must hold: if the agent knows that φ, then he necessarily knows it, and if he doesn't know that φ, then it is necessary that he doesn't know it.

Proposition 3.46 $\models \Box\mathbf{Know}(\varphi) \vee \Box\neg\mathbf{Know}(\varphi)$

Proof:
1. Let's first show that $\models \mathbf{Know}(\varphi) \supset \Box\mathbf{Know}(\varphi)$. Assume that $\langle w, a, t\rangle, g \models \mathbf{Know}(\varphi)$. Take an arbitrary w^*, such that $w \approx_t w^*$. Let's show that $\langle w^*, a, t\rangle, g \models \mathbf{Know}(\varphi)$.

$\langle w, a, t\rangle, g \models \mathbf{Know}(\varphi)$
iff for all e' such that $\langle\langle w, a, t\rangle, e'\rangle \in K$, $e', g \models \varphi$ by the definition of satisfaction
iff for all e' such that $\langle\langle w^*, a, t\rangle, e'\rangle \in K$, $e', g \models \varphi$
since by constraint 3.6, $\langle\langle w^*, a, t\rangle, e'\rangle \in K$ iff $\langle\langle w, a, t\rangle, e'\rangle \in K$
iff $\langle w^*, a, t\rangle, g \models \mathbf{Know}(\varphi)$ by the definition of satisfaction

(b) Let's now show that $\models \neg\mathbf{Know}(\varphi) \supset \Box\neg\mathbf{Know}(\varphi)$. Assume that $\langle w, a, t\rangle, g \models \neg\mathbf{Know}(\varphi)$. Take an arbitrary w^*, such that $w \approx_t w^*$.

$\langle w, a, t\rangle, g \models \neg\mathbf{Know}(\varphi)$
iff there exists an e' such that $\langle\langle w, a, t\rangle, e'\rangle \in K$, $e', g \not\models \varphi$ by the definition of satisfaction
iff there exists an e' such that $\langle\langle w^*, a, t\rangle, e'\rangle \in K$, $e', g \not\models \varphi$
since by constraint 3.6, $\langle\langle w^*, a, t\rangle, e'\rangle \in K$ iff $\langle\langle w, a, t\rangle, e'\rangle \in K$
iff $\langle w^*, a, t\rangle, g \models \neg\mathbf{Know}(\varphi)$ by the definition of satisfaction

■

For the other modal operators, the above principle is not valid, but weaker principles are. Thus for formulas of the form $\mathbf{By}(\theta^a, \varphi)$, we have that it is necessary that φ holds for agent θ^a if and only if for θ^a, φ is necessary:

Proposition 3.47 $\models \Box\mathbf{By}(\theta^a, \varphi) \equiv \mathbf{By}(\theta^a, \Box\varphi)$

Proof:
$\langle w, a, t\rangle, g \models \mathbf{By}(\theta^a, \Box\varphi)$
iff $\langle w, [\![\theta^a]\!]_{\langle w,a,t\rangle,g}, t\rangle, g \models \Box\varphi$ by the definition of satisfaction
iff for all w^* such that $w \approx_t w^*$, $\langle w^*, [\![\theta^a]\!]_{\langle w,a,t\rangle,g}, t\rangle, g \models \varphi$ by the definition of satisfaction
iff for all w^* such that $w \approx_t w^*$, $\langle w^*, [\![\theta^a]\!]_{\langle w^*,a,t\rangle,g}, t\rangle, g \models \varphi$ by lemma 3.4
iff for all w^* such that $w \approx_t w^*$, $\langle w^*, a, t\rangle, g \models \mathbf{By}(\theta^a, \varphi)$ by the definition of satisfaction
iff $\langle w, a, t\rangle, g \models \Box\mathbf{By}(\theta^a, \varphi)$ by the definition of satisfaction

■

It easily follows from this that it is possible that φ holds for θ^a if and only if for θ^a, it is possible that φ holds:

Proposition 3.48 $\models \Diamond\mathbf{By}(\theta^a, \varphi) \equiv \mathbf{By}(\theta^a, \Diamond\varphi)$

Proof:
1. $\models \neg\Box\mathbf{By}(\theta^a, \neg\varphi) \equiv \neg\mathbf{By}(\theta^a, \Box\neg\varphi)$, by proposition 3.47.
2. $\models \Diamond\neg\mathbf{By}(\theta^a, \neg\varphi) \equiv \neg\mathbf{By}(\theta^a, \neg\Diamond\varphi)$, the definition of $\Diamond$ and substitution of equivalents.
3. $\models \Diamond\mathbf{By}(\theta^a, \neg\neg\varphi) \equiv \mathbf{By}(\theta^a, \neg\neg\Diamond\varphi)$, the proposition 3.30.
4. $\models \Diamond\mathbf{By}(\theta^a, \varphi) \equiv \mathbf{By}(\theta^a, \Diamond\varphi)$, by substitution of equivalents. ■

Finally, we need to look at the interactions between historical necessity and the temporal operator **At**. First note that the **At** operator does not commute with either historical necessity or historical possibility. As the following proposition shows, we have that: (1) even if it is now necessary that φ holds at time θ^t, it need not be the case that φ is necessary at time θ^t; (2) even if φ is possible at time θ^t, it need not be the case that it is now possible that φ holds at time θ^t (3) even if φ is necessary at time θ^t, it need not be the case that it is now necessary that φ holds at time θ^t; and (4) even if it is now possible that φ holds at time θ^t, it need not be the case that φ is possible at time θ^t.

Proposition 3.49

1. $\not\models \Box\mathbf{At}(\theta^t, \varphi) \supset \mathbf{At}(\theta^t, \Box\varphi)$

2. $\not\models \mathbf{At}(\theta^t, \Diamond\varphi) \supset \Diamond\mathbf{At}(\theta^t, \varphi)$

3. $\not\models \mathbf{At}(\theta^t, \Box\varphi) \supset \Box\mathbf{At}(\theta^t, \varphi)$

4. $\not\models \Diamond\mathbf{At}(\theta^t, \varphi) \supset \mathbf{At}(\theta^t, \Diamond\varphi)$

Proof:
We construct a counter model for the formulas. Let $\mathcal{W} = \{\mathsf{w_0}, \mathsf{w_1}\}$ and $\mathcal{T} = \{\mathsf{t_0}, \mathsf{t_1}\}$. Let $\mathsf{t_0} \prec \mathsf{t_1}$, $\approx_{\mathsf{t_0}} = \mathcal{W}^2$, and $\approx_{\mathsf{t_1}} = \{\langle \mathsf{w_0}, \mathsf{w_0}\rangle, \langle \mathsf{w_1}, \mathsf{w_1}\rangle\}$. Finally, let $\langle \mathsf{w_0}, \mathsf{a}, \mathsf{t_1}\rangle, \mathsf{g} \models \mathrm{R}$, $\langle \mathsf{w_1}, \mathsf{a}, \mathsf{t_1}\rangle, \mathsf{g} \not\models \mathrm{R}$, $\mathsf{g}(t_0) = \mathsf{t_0}$ and $\mathsf{g}(t_1) = \mathsf{t_1}$.
1. We have $\langle \mathsf{w_0}, \mathsf{a}, \mathsf{t_1}\rangle, \mathsf{g} \models \Box\mathbf{At}(t_0, \mathbf{At}(t_1, \mathrm{R}))$, and yet $\langle \mathsf{w_0}, \mathsf{a}, \mathsf{t_1}\rangle, \mathsf{g} \not\models \mathbf{At}(t_0, \Box\mathbf{At}(t_1, \mathrm{R}))$; so 1 is not valid.
2. We have $\langle \mathsf{w_1}, \mathsf{a}, \mathsf{t_1}\rangle, \mathsf{g} \models \mathbf{At}(t_0, \Diamond\mathbf{At}(t_1, \mathrm{R}))$, and yet $\langle \mathsf{w_1}, \mathsf{a}, \mathsf{t_1}\rangle, \mathsf{g} \not\models \Diamond\mathbf{At}(t_0, \mathbf{At}(t_1, \mathrm{R}))$; so 2 is not valid.
3. We have that $\langle \mathsf{w_0}, \mathsf{a}, \mathsf{t_0}\rangle, \mathsf{g} \models \mathbf{At}(t_1, \Box\mathrm{R})$, and yet $\langle \mathsf{w_0}, \mathsf{a}, \mathsf{t_0}\rangle, \mathsf{g} \not\models \Box\mathbf{At}(t_1, \mathrm{R})$; so 3 is not valid.
4. We have that $\langle \mathsf{w_1}, \mathsf{a}, \mathsf{t_0}\rangle, \mathsf{g} \models \Diamond\mathbf{At}(t_1, \mathrm{R})$, and yet $\langle \mathsf{w_1}, \mathsf{a}, \mathsf{t_0}\rangle, \mathsf{g} \not\models \mathbf{At}(t_1, \Diamond\mathrm{R})$; so 4 is not valid either. ■

However, if we look at various restricted forms of the **At** operator, we see that some useful principles do emerge. We first examine the interactions of historical necessity and possibility with the tense logic operators, starting with the ones that talk about the past. The following proposition shows that it must be the case that if at some past time φ was necessary, then it is now necessary that φ held at some past time.

Proposition 3.50 $\models$ **SomePast**$(\Box\varphi) \supset \Box$**SomePast**$(\varphi)$

Proof:
Assume that $\langle w, a, t\rangle, g \models$ **SomePast**$(\Box\varphi)$, that is, that there exists a t_p, such that $t_p \preceq t$ and for all w_p^* such that $w \approx_{t_p} w_p^*$, $\langle w_p^*, a, t_p\rangle, g \models \varphi$. Take an arbitrary w^*, such that $w \approx_t w^*$. By constraint 3.5, it must be the case that $w \approx_{t_p} w^*$. By the assumption, it must also be the case that $\langle w^*, a, t_p\rangle, g \models \varphi$. So it must be the case that $\langle w^*, a, t\rangle, g \models$ **SomePast**(φ). And since w^* is an arbitrary world, such that $w \approx_t w^*$, it must also be the case that $\langle w, a, t\rangle, g \models \Box$**SomePast**$(\varphi)$. ■

This is easily shown to imply that $\models \Diamond$**AllPast**$(\varphi) \supset$ **AllPast**$(\Diamond\varphi)$. We also have that if it is now possible that φ held at some past time, then at some past time, φ must have been possible.

Proposition 3.51 $\models \Diamond$**SomePast**$(\varphi) \supset$ **SomePast**$(\Diamond\varphi)$

Proof:
Assume that $\langle w, a, t\rangle, g \models \Diamond$**SomePast**$(\varphi)$, that is, that there exist a w^* and a t_p, such that $w \approx_t w^*$, $t_p \preceq t$, and $\langle w^*, a, t_p\rangle, g \models \varphi$. By constraint 3.5, it must be the case that $w \approx_{t_p} w^*$. So it must be the case that there exist a t_p and a w^*, such that $t_p \preceq t$, $w \approx_t w^*$, and $\langle w^*, a, t_p\rangle, g \models \varphi$. This means that $\langle w, a, t\rangle, g \models$ **SomePast**$(\Diamond\varphi)$. ■

Note that this trivially implies that $\models$ **AllPast**$(\Box\varphi) \supset \Box$**AllPast**$(\varphi)$. On the other hand, we have that (1) even if it is now necessary that φ held at some past time, it need not be the case that φ was necessary at some past time, and (2) even if φ was possible at some past time, it need not be the case that it is now possible that φ held at some past time:

Proposition 3.52

1. $\not\models \Box$**SomePast**$(\varphi) \supset$ **SomePast**$(\Box\varphi)$

2. $\not\models$ **SomePast**$(\Diamond\varphi) \supset \Diamond$**SomePast**$(\varphi)$

Proof:
We use the structure specified in the proof of proposition 3.49 to show that the formulas are falsifiable.
1. We have $\langle w_0, a, t_1\rangle, g \models \Box$**SomePast**$(\mathbf{At}(t_1, \mathrm{R}))$,
and yet that $\langle w_0, a, t_1\rangle, g \not\models$ **SomePast**$(\Box\mathbf{At}(t_1, \mathrm{R}))$; so 1 is not valid.
2. We have $\langle w_1, a, t_1\rangle, g \models$ **SomePast**$(\Diamond\mathbf{At}(t_1, \mathrm{R}))$,
and yet $\langle w_1, a, t_1\rangle, g \not\models \Diamond$**SomePast**$(\mathbf{At}(t_1, \mathrm{R}))$; so 2 is not valid. ■

Note that as a consequence of 1 we have that $\not\models$ **AllPast**($\Diamond\varphi$) $\supset$ $\Diamond$**AllPast**(φ). Similarly, 2 implies that $\not\models$ $\Box$**AllPast**(φ) $\supset$ **AllPast**($\Box\varphi$).

Now let's look at tense logic claims about the future. All we have in this case is that if at some future time, φ is possible, then it must now be possible that φ will hold at some later time.

Proposition 3.53 $\models$ **SomeFut**($\Diamond\varphi$) $\supset$ $\Diamond$**SomeFut**(φ)

Proof:
Assume that $\langle w, a, t\rangle, g \models$ **SomeFut**($\Diamond\varphi$), that is, that there exist a t_f and a w^*, such that $t \preceq t_f$, $w \approx_{t_f} w^*$, and $\langle w^*, a, t_f\rangle, g \models \varphi$. By constraint 3.5, it must be the case that $w \approx_t w^*$. So it must be the case that there exist a w^* and a t_f, such that $w \approx_t w^*$, $t \preceq t_f$, and $\langle w^*, a, t_f\rangle, g \models \varphi$. This means that $\langle w, a, t\rangle, g \models \Diamond$**SomeFut**($\varphi$). ■

This trivially yields that $\models$ $\Box$**AllFut**(φ) $\supset$ **AllFut**($\Box\varphi$). On the other hand we have that: (1) even if at some point in the future φ will be necessary, it need not be the case that it is now necessary that φ will hold; (2) even if it is currently possible that φ will later hold, it need not be the case that φ will be possible later; and (3) even if it is currently necessary that φ will hold at some future time, this does not mean that φ will be necessary at some future time.

Proposition 3.54

1. $\not\models$ **SomeFut**($\Box\varphi$) $\supset$ $\Box$**SomeFut**(φ)

2. $\not\models$ $\Diamond$**SomeFut**(φ) $\supset$ **SomeFut**($\Diamond\varphi$)

3. $\not\models$ $\Box$**SomeFut**(φ) $\supset$ **SomeFut**($\Box\varphi$)

Proof:
1. Consider the structure specified in the proof of proposition 3.49. We have $\langle w_0, a, t_0\rangle, g \models$ **SomeFut**($\Box$R), and yet $\langle w_0, a, t_0\rangle, g \not\models$ $\Box$**SomeFut**(R); so 1 is not valid.
2. Again, consider the structure specified in the proof of proposition 3.49. We have $\langle w_1, a, t_0\rangle, g \models \Diamond$**SomeFut**(R), and yet $\langle w_1, a, t_0\rangle, g \not\models$ **SomeFut**($\Diamond$R); so 2 is not valid.
3. Let $\mathcal{W} = \{w_0, w_1\}$ and $\mathcal{T} = \{t_0, t_1, t_2, t_3\}$. Let $t_0 \prec t_1 \prec t_2 \prec t_3$, $\approx_{t_0} = \approx_{t_1} = \mathcal{W}^2$, and $\approx_{t_2} = \approx_{t_3} = \{\langle w_0, w_0\rangle, \langle w_1, w_1\rangle\}$. Finally, let $\Delta = \{\langle d, \langle w_0, a, t_1\rangle, t_2\rangle, \langle d, \langle w_1, a, t_2\rangle, t_3\rangle\}$ and $g(d) = d$. Then, we have that $\langle w_0, a, t_0\rangle, g \models$ $\Box$**SomeFut**(**DoesNext**(d)), and yet $\langle w_0, a, t_0\rangle, g \not\models$ **SomeFut**($\Box$**DoesNext**(d)); so 3 is not valid. ■

Note that 1 implies that $\not\models$ $\Diamond$**AllFut**(φ) $\supset$ **AllFut**($\Diamond\varphi$), 2 implies that $\not\models$ **AllFut**($\Box\varphi$) $\supset$ $\Box$**AllFut**(φ), and 3 implies that $\not\models$ **AllFut**($\Diamond\varphi$) $\supset$ $\Diamond$**AllFut**(φ).

We have already seen that whether an action will be done next or not is historically contingent (proposition 3.45). However, what has already been done is necessary. More generally, we have that if θ^d has just been done and that φ was necessary beforehand, then it must be the case that it is now necessary that θ^d has just been done and that φ held beforehand:

Proposition 3.55 $\models$ **DoneWhen**$(\theta^d, \Box\varphi) \supset \Box$**DoneWhen**$(\theta^d, \varphi)$

Proof:
1. Assume that $\langle w, a, t\rangle, g \models$ **DoneWhen**$(\theta^d, \Box\varphi)$. This means that there exists a t_s, such that $\Delta([\![\theta^d]\!]_{\langle w,a,t_s\rangle,g}, \langle w, a, t_s\rangle, t)$ and for all w_s^* such that $w \approx_{t_s} w_s^*$, $\langle w_s^*, a, t_s\rangle, g \models \varphi$.
2. Suppose that $\langle w, a, t\rangle, g \not\models \Box$**DoneWhen**$(\theta^d, \varphi)$, that is, that there exists a w^* such that $w \approx_t w^*$, for all t_s', either it is not the case that $\Delta([\![\theta^d]\!]_{\langle w^*,a,t_s'\rangle,g}, \langle w^*, a, t_s'\rangle, t)$, or $\langle w^*, a, t_s'\rangle, g \not\models \varphi$.
3. By 1, 2, and lemma 3.4, we have that $[\![\theta^d]\!]_{\langle w^*,a,t_s\rangle,g} = [\![\theta^d]\!]_{\langle w,a,t_s\rangle,g}$. So by 1, we must have that $\Delta([\![\theta^d]\!]_{\langle w^*,a,t_s\rangle,g}, \langle w, a, t_s\rangle, t)$. Then by constraint 3.6, we must also have that $\Delta([\![\theta^d]\!]_{\langle w^*,a,t_s\rangle,g}, \langle w^*, a, t_s\rangle, t)$.
4. Thus, by this and 2, we must have that $\langle w^*, a, t_s\rangle, g \not\models \varphi$.
5. By 1 and constraint 3.3, we must have that $t_s \preceq t$. Since $w \approx_t w^*$, by constraint 3.5, it must also be the case that $w \approx_{t_s} w^*$.
6. By 4 and 5, we have that $w \approx_{t_s} w^*$ and $\langle w^*, a, t_s\rangle, g \not\models \varphi$. This contradicts 1. ■

We also have that if it is now possible that θ^d has just been done and that φ held beforehand, then it must be the case that θ^d has just been done and that φ was possible beforehand:

Proposition 3.56 $\models \Diamond$**DoneWhen**$(\theta^d, \varphi) \supset$ **DoneWhen**$(\theta^d, \Diamond\varphi)$

Proof:
1. Assume that $\langle w, a, t\rangle, g \models \Diamond$**DoneWhen**$(\theta^d, \varphi)$. This means that there exists a w^* such that $w \approx_t w^*$ and there exists a t_s, such that $\Delta([\![\theta^d]\!]_{\langle w^*,a,t_s\rangle,g}, \langle w^*, a, t_s\rangle, t)$ and $\langle w^*, a, t_s\rangle, g \models \varphi$.
2. Since $w \approx_t w^*$, by lemma 3.4, we have that $[\![\theta^d]\!]_{\langle w^*,a,t_s\rangle,g} = [\![\theta^d]\!]_{\langle w,a,t_s\rangle,g}$. So by 1, we must have that $\Delta([\![\theta^d]\!]_{\langle w,a,t_s\rangle,g}, \langle w^*, a, t_s\rangle, t)$. Then by constraint 3.6, we must also have that $\Delta([\![\theta^d]\!]_{\langle w,a,t_s\rangle,g}, \langle w, a, t_s\rangle, t)$.
3. By constraint 3.3, we must have that $t_s \preceq t$. Since $w \approx_t w^*$, by constraint 3.5, it must also be the case that $w \approx_{t_s} w^*$. Thus by 1, we have that there exists a w^* such that $w \approx_{t_s} w^*$ and $\langle w^*, a, t_s\rangle, g \models \varphi$, that is, that $\langle w, a, t_s\rangle, g \models \Diamond\varphi$.
4. Thus, by 1, 2, and 3, we have that there exists a t_s, such that $\Delta([\![\theta^d]\!]_{\langle w,a,t_s\rangle,g}, \langle w, a, t_s\rangle, t)$ and $\langle w, a, t_s\rangle, g \models \Diamond\varphi$. ■

Note that since $\models$ **Done**$(\theta^d) \equiv$ **DoneWhen**$(\theta^d,$ **True**$)$ (we take **True** to stand for $R \vee \neg R$), it follows easily from the last two propositions that $\models \Box$**Done**$(\theta^d) \vee \Box\neg$**Done**(θ^d).

On the other hand, we have that (1) even if it is now necessary that θ^d has just been done and that φ held beforehand, it need not be the case that θ^d has just been done and that φ necessarily held beforehand, and (2) even if θ^d has just been done and φ was possible beforehand, it need not be the case that it is now possible that θ^d has just been done and that φ held beforehand:

Proposition 3.57

1. $\not\models \Box\mathbf{DoneWhen}(\theta^d, \varphi) \supset \mathbf{DoneWhen}(\theta^d, \Box\varphi)$

2. $\not\models \mathbf{DoneWhen}(\theta^d, \Diamond\varphi) \supset \Diamond\mathbf{DoneWhen}(\theta^d, \varphi)$

Proof:
Consider a structure as in the proof of proposition 3.49, with the additional requirements that $\Delta(\mathsf{d}, \langle \mathsf{w}_0, \mathsf{a}, \mathsf{t}_0\rangle, \mathsf{t}_1)$ and $\Delta(\mathsf{d}, \langle \mathsf{w}_1, \mathsf{a}, \mathsf{t}_0\rangle, \mathsf{t}_1)$. Also let $\mathsf{g}(t_1) = \mathsf{t}_1$ and $\mathsf{g}(d) = \mathsf{d}$.
1. In this structure we have that $\langle \mathsf{w}_0, \mathsf{a}, \mathsf{t}_1\rangle, \mathsf{g} \models \Box\mathbf{DoneWhen}(d, \mathbf{At}(t_1, \mathrm{R}))$, and yet $\langle \mathsf{w}_0, \mathsf{a}, \mathsf{t}_1\rangle, \mathsf{g} \not\models \mathbf{DoneWhen}(d, \Box\mathbf{At}(t_1, \mathrm{R}))$; so 1 is not valid.
2. It is also the case that $\langle \mathsf{w}_1, \mathsf{a}, \mathsf{t}_1\rangle, \mathsf{g} \models \mathbf{DoneWhen}(d, \Diamond\mathbf{At}(t_1, \mathrm{R}))$, while $\langle \mathsf{w}_1, \mathsf{a}, \mathsf{t}_1\rangle, \mathsf{g} \not\models \Diamond\mathbf{DoneWhen}(d, \mathbf{At}(t_1, \mathrm{R}))$; so 2 is not valid. ■

Now that we have proven that the historical necessity and possibility operators obey all the above principles, we are able to establish the general result set out earlier. The following proposition says that if φ is future-blind, then either it is necessary or its negation is:

Proposition 3.58 $\models \Box\varphi \vee \Box\neg\varphi$, provided that φ is future-blind,

Proof:
We prove this by structural induction.
(1) In case φ is atomic. Then by the proviso it cannot be of the form $\mathbf{Does}(\theta^d, \theta^t)$, so by proposition 3.44, the statement must hold.
(2) In case $\varphi = \neg\varphi'$. Assume that the proviso holds for φ. Then it must also hold for φ'. Thus by the induction hypothesis, $\models \Box\varphi' \vee \Box\neg\varphi'$. By substitution of equivalents and propositional reasoning, it follows that $\models \Box\neg\varphi' \vee \Box\neg\neg\varphi'$.
(3) In case $\varphi = (\varphi_1 \supset \varphi_2)$. Assume that the proviso holds for φ. Then it must also hold for φ_1 and φ_2. Thus by the induction hypothesis, $\models \Box\varphi_1 \vee \Box\neg\varphi_1$ and $\models \Box\varphi_2 \vee \Box\neg\varphi_2$. We must show that $\models \Box(\varphi_1 \supset \varphi_2) \vee \Box\neg(\varphi_1 \supset \varphi_2)$.
(a) If $\mathsf{e}, \mathsf{g} \models \varphi_2$. Then by the induction hypothesis, we must have that $\mathsf{e}, \mathsf{g} \models \Box\varphi_2$. This implies that $\mathsf{e}, \mathsf{g} \models \Box(\varphi_1 \supset \varphi_2)$.
(b) If on the other hand $\mathsf{e}, \mathsf{g} \models \neg\varphi_2$. By the induction hypothesis, we have that $\mathsf{e}, \mathsf{g} \models \Box\neg\varphi_2$. We must consider whether or not φ_1 is satisfied by e and g. If it is, then by the induction hypothesis, we have that $\mathsf{e}, \mathsf{g} \models \Box\varphi_1$, which implies that $\mathsf{e}, \mathsf{g} \models \Box\neg(\varphi_1 \supset \varphi_2)$. If it is not, then by the induction hypothesis, we have that $\mathsf{e}, \mathsf{g} \models \Box\neg\varphi_1$, which implies that $\mathsf{e}, \mathsf{g} \models \Box(\varphi_1 \supset \varphi_2)$.
(4) In case $\varphi = \forall v\varphi'$. Assume that the proviso holds for φ. Then it must also hold for φ'. Thus by the induction hypothesis, $\models \Box\varphi' \vee \Box\neg\varphi'$. We must show that $\models \Box\forall v\varphi' \vee \Box\neg\forall v\varphi'$. Consider the following cases:
(a) If $\mathsf{e}, \mathsf{g} \models \forall v\varphi'$. This means that for all x in the appropriate domain, $\mathsf{e}, \mathsf{g}\{v \mapsto \mathsf{x}\} \models \varphi'$. By the induction hypothesis, we must have that for all x in the appropriate domain, $\mathsf{e}, \mathsf{g}\{v \mapsto \mathsf{x}\} \models \Box\varphi'$, that is, that $\langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle, \mathsf{g} \models \forall v\Box\varphi'$. By proposition 3.42, this implies that $\langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle, \mathsf{g} \models \Box\forall v\varphi'$.
(b) If $\mathsf{e}, \mathsf{g} \models \neg\forall v\varphi'$. This means that there exists an x in the appropriate domain, such that $\mathsf{e}, \mathsf{g}\{v \mapsto \mathsf{x}\} \models \neg\varphi'$. By the induction hypothesis, we must have that there exists an x in the appropriate domain, such that $\mathsf{e}, \mathsf{g}\{v \mapsto \mathsf{x}\} \models \Box\neg\varphi'$, that is, that $\mathsf{e}, \mathsf{g} \models \exists v\Box\neg\varphi'$. By

proposition 3.43, this implies that $\mathbf{e},\mathbf{g} \models \Box\exists v\neg\varphi'$, and thus that $\mathbf{e},\mathbf{g} \models \Box\neg\forall v\varphi'$.
(5) If $\varphi = \mathbf{Know}(\varphi')$, then by proposition 3.46, the statement holds.
(6) In case $\varphi = \mathbf{By}(\theta^a, \varphi')$. Assume that the proviso holds for φ. Then it must also hold for φ'. Thus by the induction hypothesis, $\models \Box\varphi' \vee \Box\neg\varphi'$. We must show that $\models \Box\mathbf{By}(\theta^a, \varphi') \vee \Box\neg\mathbf{By}(\theta^a, \varphi')$. Consider the following cases:
(a) If $\langle \mathrm{w}, [\![\theta^a]\!]_{\langle \mathrm{w,a,t}\rangle,\mathrm{g}}, \mathrm{t}\rangle, \mathrm{g} \models \varphi'$. Then by the induction hypothesis, it must be the case that $\langle \mathrm{w}, [\![\theta^a]\!]_{\langle \mathrm{w,a,t}\rangle,\mathrm{g}}, \mathrm{t}\rangle, \mathrm{g} \models \Box\varphi'$, that is, that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \mathbf{By}(\theta^a, \Box\varphi')$. By proposition 3.47, this implies that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \Box\mathbf{By}(\theta^a, \varphi')$.
(b) If on the other hand $\langle \mathrm{w}, [\![\theta^a]\!]_{\langle \mathrm{w,a,t}\rangle,\mathrm{g}}, \mathrm{t}\rangle, \mathrm{g} \models \neg\varphi'$. Then by the induction hypothesis, we must have that $\langle \mathrm{w}, [\![\theta^a]\!]_{\langle \mathrm{w,a,t}\rangle,\mathrm{g}}, \mathrm{t}\rangle, \mathrm{g} \models \Box\neg\varphi'$, that is, that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \mathbf{By}(\theta^a, \Box\neg\varphi')$. By proposition 3.47, this implies that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \Box\mathbf{By}(\theta^a, \neg\varphi')$. By propositions 3.30, 3.37, and 3.40, this implies that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \Box\neg\mathbf{By}(\theta^a, \varphi')$.
(7) In case $\varphi = \Box\varphi'$. Assume that the proviso holds for φ. Then it must also hold for φ'. Thus by the induction hypothesis, $\models \Box\varphi' \vee \Box\neg\varphi'$. We must show that $\models \Box\Box\varphi' \vee \Box\neg\Box\varphi'$. Consider the following cases:
(a) If $\mathbf{e},\mathbf{g} \models \Box\varphi'$. Then by proposition 3.41, $\mathbf{e},\mathbf{g} \models \Box\Box\varphi'$.
(b) If $\mathbf{e},\mathbf{g} \models \neg\Box\varphi'$. By the definition of $\Diamond$ and substitution of equivalents, this means that $\mathbf{e},\mathbf{g} \models \Diamond\neg\varphi'$. By proposition 3.39, it must then be the case that $\mathbf{e},\mathbf{g} \models \Box\Diamond\neg\varphi'$. By the definition of $\Diamond$ and substitution of equivalents, this means that $\mathbf{e},\mathbf{g} \models \Box\neg\Box\varphi'$.
(8) In case $\varphi = \mathbf{SomePast}(\varphi')$. Assume that the proviso holds for φ. Then it must also hold for φ'. Thus by the induction hypothesis, $\models \Box\varphi' \vee \Box\neg\varphi'$. We must show that $\models \Box\mathbf{SomePast}(\varphi') \vee \Box\neg\mathbf{SomePast}(\varphi')$. Consider the following cases:
(a) If $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \mathbf{SomePast}(\varphi')$. This means that there exists a t_p, such that $\mathrm{t}_p \prec \mathrm{t}$ and $\langle \mathrm{w,a,t}_p\rangle, \mathrm{g} \models \varphi'$. By the induction hypothesis, it must then be the case that there exists a t_p, such that $\mathrm{t}_p \prec \mathrm{t}$ and $\langle \mathrm{w,a,t}_p\rangle, \mathrm{g} \models \Box\varphi'$, that is, that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \mathbf{SomePast}(\Box\varphi')$. Then by proposition 3.50, we must thus have that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \Box\mathbf{SomePast}(\varphi')$.
(b) If $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \neg\mathbf{SomePast}(\varphi')$. This means that for all t_p such that $\mathrm{t}_p \prec \mathrm{t}$, it is the case that $\langle \mathrm{w,a,t}_p\rangle, \mathrm{g} \models \neg\varphi'$. By the induction hypothesis, it must then be the case that for all t_p such that $\mathrm{t}_p \prec \mathrm{t}$, $\langle \mathrm{w,a,t}_p\rangle, \mathrm{g} \models \Box\neg\varphi'$. By the definition of $\Diamond$ and substitution of equivalents, this means that for all t_p such that $\mathrm{t}_p \prec \mathrm{t}$, $\langle \mathrm{w,a,t}_p\rangle, \mathrm{g} \models \neg\Diamond\varphi'$, which means that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \neg\mathbf{SomePast}(\Diamond\varphi')$. Then by proposition 3.51, we must thus have that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \neg\Diamond\mathbf{SomePast}(\varphi')$. And by the definition of $\Diamond$ and substitution of equivalents, this means that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \Box\neg\mathbf{SomePast}(\varphi')$.
(9) In case $\varphi = \mathbf{DoneWhen}(\theta^d, \varphi')$. Assume that the proviso holds for φ. Then it must also hold for φ'. Thus by the induction hypothesis, $\models \Box\varphi' \vee \Box\neg\varphi'$. We must show that $\models \Box\mathbf{DoneWhen}(\theta^d, \varphi') \vee \Box\neg\mathbf{DoneWhen}(\theta^d, \varphi')$. Consider the following cases:
(a) If $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \mathbf{DoneWhen}(\theta^d, \varphi')$. This means that there exists a t_s, such that $\Delta([\![\theta^d]\!]_{\langle \mathrm{w,a,t}_s\rangle,\mathrm{g}}, \langle \mathrm{w,a,t}_s\rangle, \mathrm{t})$ and $\langle \mathrm{w,a,t}_s\rangle, \mathrm{g} \models \varphi'$. By the induction hypothesis, it must then be the case that there exists a t_s, such that $\Delta([\![\theta^d]\!]_{\langle \mathrm{w,a,t}_s\rangle,\mathrm{g}}, \langle \mathrm{w,a,t}_s\rangle, \mathrm{t})$ and $\langle \mathrm{w,a,t}_s\rangle, \mathrm{g} \models \Box\varphi'$, that is, that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \mathbf{DoneWhen}(\theta^d, \Box\varphi')$. Then by proposition 3.55, we must thus have that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \Box\mathbf{DoneWhen}(\theta^d, \varphi')$.
(b) If $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \neg\mathbf{DoneWhen}(\theta^d, \varphi')$. This means that for all t_s, either it is not the case that $\Delta([\![\theta^d]\!]_{\langle \mathrm{w,a,t}_s\rangle,\mathrm{g}}, \langle \mathrm{w,a,t}_s\rangle, \mathrm{t})$ or $\langle \mathrm{w,a,t}_s\rangle, \mathrm{g} \models \neg\varphi'$. By the induction hypothesis, it must then be the case that for all t_s, either it is not the case that $\Delta([\![\theta^d]\!]_{\langle \mathrm{w,a,t}_s\rangle,\mathrm{g}}, \langle \mathrm{w,a,t}_s\rangle, \mathrm{t})$ or $\langle \mathrm{w,a,t}_s\rangle, \mathrm{g} \models \Box\neg\varphi'$. By the definition of $\Diamond$ and substitution of equivalents, this means that for all t_s, either it is not the case that $\Delta([\![\theta^d]\!]_{\langle \mathrm{w,a,t}_s\rangle,\mathrm{g}}, \langle \mathrm{w,a,t}_s\rangle, \mathrm{t})$ or $\langle \mathrm{w,a,t}_s\rangle, \mathrm{g} \models \neg\Diamond\varphi'$, which means that $\langle \mathrm{w,a,t}\rangle, \mathrm{g} \models \neg\mathbf{DoneWhen}(\theta^d, \Diamond\varphi')$. Then by proposition 3.56, we

must thus have that $\langle \mathrm{w}, \mathrm{a}, \mathrm{t}\rangle, \mathrm{g} \models \neg\Diamond\mathbf{DoneWhen}(\theta^d, \varphi')$. And by the definition of $\Diamond$ and substitution of equivalents, this means that $\langle \mathrm{w}, \mathrm{a}, \mathrm{t}\rangle, \mathrm{g} \models \Box\neg\mathbf{DoneWhen}(\theta^d, \varphi')$. ■

From this, it follows straightforwardly that if φ is future-blind, then it is necessary if and only if it is true:

Proposition 3.59 $\models \Box\varphi \equiv \varphi$, provided that φ is future-blind

Proof:
Proposition 3.38 verifies that $\models \Box\varphi \supset \varphi$. Let's now show that the converse must hold. Assume that $\mathrm{e}, \mathrm{g} \models \varphi$. Assume also that the proviso holds. Then by proposition 3.58, we must have that $\mathrm{e}, \mathrm{g} \models \Box\varphi \vee \Box\neg\varphi$. Suppose that $\mathrm{e}, \mathrm{g} \models \Box\neg\varphi$; by proposition 3.38, this implies that $\mathrm{e}, \mathrm{g} \models \neg\varphi$, which contradict our earlier assumption. So we must have that $\mathrm{e}, \mathrm{g} \models \Box\varphi$. ■

3.5.6 PhyPoss, AfterNec, and Res

In this section, we discuss various properties of the derived operators **AfterNec**, **PhyPoss**, and **Res** (which are defined on page 60). The propositions stated will turn out to be very useful tools for proving that agents are able to achieve goals in later chapters; they allow us to reason at a more abstract level in developing proofs of ability.

The first three propositions show essentially that the action parameter of the **AfterNec**, **PhyPoss**, and **Res** operators is transparent, that is, that action terms that have the same denotation can be substituted in the action parameter of these operators without affecting the truth-value of the formula. Thus, the following proposition says that if θ_1^d stands for the same action as θ_2^d, then φ must hold after the agent does action θ_1^d if and only if φ must hold after the agent does θ_2^d:

Proposition 3.60 $\models \theta_1^d = \theta_2^d \supset (\mathbf{AfterNec}(\theta_1^d, \varphi) \equiv \mathbf{AfterNec}(\theta_2^d, \varphi))$

Proof:
Assume that $\mathrm{e}, \mathrm{g} \models \theta_1^d = \theta_2^d$. We only show that $\mathrm{e}, \mathrm{g} \models \mathbf{AfterNec}(\theta_1^d, \varphi) \supset \mathbf{AfterNec}(\theta_2^d, \varphi)$; since $=$ is symmetric, it trivially follows that if the statement holds in one direction, it must also hold in the other. So assume that $\mathrm{e}, \mathrm{g} \models \mathbf{AfterNec}(\theta_1^d, \varphi)$, that is, that for all w^* and t_e, such that $\mathrm{w} \approx_{\mathrm{t}} \mathrm{w}^*$, $\mathrm{e}^*, \mathrm{g}_{\mathrm{t}_e} \models \mathbf{Does}(\theta_1^d, v^t) \supset \mathbf{At}(v^t, \varphi)$, where v^t is a variable that does not occur free in φ, $\mathrm{e}^* = \langle \mathrm{w}^*, \mathrm{a}, \mathrm{t}\rangle$, and $\mathrm{g}_{\mathrm{t}_e} = \{v^t \mapsto \mathrm{t}_e\}$. By lemma 3.4, it must be that case that $[\![\theta_1^*]\!]_{\mathrm{e}^*, \mathrm{g}_{\mathrm{t}_e}} = [\![\theta_2^*]\!]_{\mathrm{e}^*, \mathrm{g}_{\mathrm{t}_e}}$. Thus, it follows that $\mathrm{e}^*, \mathrm{g}_{\mathrm{t}_e} \models \mathbf{Does}(\theta_2^d, v^t) \supset \mathbf{At}(v^t, \varphi)$. ■

The next proposition says that the same holds for **PhyPoss**, that is, that if θ_1^d stands for the same action as θ_2^d, then it is possible for the agent to do θ_1^d next if and only if it is possible for him to do θ_2^d next:

Proposition 3.61 $\models \theta_1^d = \theta_2^d \supset (\mathbf{PhyPoss}(\theta_1^d) \equiv \mathbf{PhyPoss}(\theta_2^d))$

The proof is similar to that of proposition 3.60. From these two results, it follows trivially that **Res** is also transparent in its action parameter. So we get that if θ_1^d stands for the same action as θ_2^d, then the agent's doing action θ_1^d next must result in φ holding if and only if his doing θ_2^d next must also result in φ:

Proposition 3.62 $\models \theta_1^d = \theta_2^d \supset (\mathbf{Res}(\theta_1^d, \varphi) \equiv \mathbf{Res}(\theta_2^d, \varphi))$

The next two propositions assert that **AfterNec** and **Res** distribute over implication (i.e., the K axiom holds for them). The first says that if $\varphi_1 \supset \varphi_2$ must hold after the agent does action θ^d, and if φ_1 must hold after he does θ^d, then φ_2 must also hold after he does θ^d:

Proposition 3.63 $\models \mathbf{AfterNec}(\theta^d, \varphi_1 \supset \varphi_2) \supset (\mathbf{AfterNec}(\theta^d, \varphi_1) \supset \mathbf{AfterNec}(\theta^d, \varphi_2))$.

Proof:
We prove the result by contradiction.
1. Assume that $\mathrm{e}, \mathrm{g} \models \mathbf{AfterNec}(\theta^d, \varphi_1 \supset \varphi_2)$ and $\mathrm{e}, \mathrm{g} \models \mathbf{AfterNec}(\theta^d, \varphi_1)$, where $\mathrm{e} = \langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle$. This means that for all w^* and t_e, such that $\mathrm{w} \approx_{\mathrm{t}} \mathrm{w}^*$,

$$\langle \mathrm{w}^*, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}\{v^t \mapsto \mathrm{t}_e\} \models \mathbf{Does}(\theta^d, v^t) \supset \mathbf{At}(v^t, \varphi_1 \supset \varphi_2)$$
$$\text{and}$$
$$\langle \mathrm{w}^*, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}\{v^t \mapsto \mathrm{t}_e\} \models \mathbf{Does}(\theta^d, v^t) \supset \mathbf{At}(v^t, \varphi_1)$$

where v^t does not occur free in φ_1 and φ_2.
2. Assume that $\mathrm{e}, \mathrm{g} \not\models \mathbf{AfterNec}(\theta^d, \varphi_2)$, that is, that

$$\mathrm{e}, \mathrm{g} \not\models \Box \forall v^t (\mathbf{Does}(\theta^d, v^t) \supset \mathbf{At}(v^t, \varphi_2))$$

This means that there exist w_f and t_f, such that $\mathrm{w} \approx_{\mathrm{t}} \mathrm{w}_f$ and $\langle \mathrm{w}_f, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g}_{\mathrm{t}_f} \models \mathbf{Does}(\theta^d, v^t)$ and $\langle \mathrm{w}_f, \mathrm{a}, \mathrm{t}_f \rangle, \mathrm{g}_{\mathrm{t}_f} \not\models \varphi_2$, where $\mathrm{g}_{\mathrm{t}_f} = \{v^t \mapsto \mathrm{t}_f\}$.
3. Combining this with 1, we obtain that $\langle \mathrm{w}_f, \mathrm{a}, \mathrm{t}_f \rangle, \mathrm{g}_{\mathrm{t}_f} \models \varphi_1 \supset \varphi_2$ and $\langle \mathrm{w}_f, \mathrm{a}, \mathrm{t}_f \rangle, \mathrm{g}_{\mathrm{t}_f} \models \varphi_1$. Therefore, we must also have $\langle \mathrm{w}_f, \mathrm{a}, \mathrm{t}_f \rangle, \mathrm{g}_{\mathrm{t}_f} \models \varphi_2$, which contradicts 2. ∎

From this, it trivially follows that the same property holds for **Res**, that is, that if the agent's doing action θ^d next must result in $\varphi_1 \supset \varphi_2$ holding and if his doing θ^d next must result in φ_1, then his doing θ^d next must also result in φ_2:

Proposition 3.64 $\models \mathbf{Res}(\theta^d, \varphi_1 \supset \varphi_2) \supset (\mathbf{Res}(\theta^d, \varphi_1) \supset \mathbf{Res}(\theta^d, \varphi_2))$

The necessitation rule also holds for **AfterNec**, as shown by the following proposition. It says that if φ is valid, then it must be the case that φ must hold afterwards if the agent does action θ^d next:

Proposition 3.65 If $\models \varphi$, then $\models \mathbf{AfterNec}(\theta^d, \varphi)$

Proof:
We prove the result by contradiction.
1. Assume that $\models \varphi$.
2. Assume that $\not\models$ **AfterNec**(θ^d, φ), that is, that $\not\models \Box\forall v^t(\mathbf{Does}(\theta^d, v^t) \supset \mathbf{At}(v^t, \varphi))$, where v^t is not free in φ. This means that (in some structure) there exist $\mathrm{e} = \langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle$ and g, such that $\mathrm{e}, \mathrm{g} \not\models \Box\forall v^t(\mathbf{Does}(\theta^d, v^t) \supset \mathbf{At}(v^t, \varphi))$. Thus, there exist w^* and t_e, such that $\mathrm{w} \approx_\mathrm{t} \mathrm{w}^*$ and $\mathrm{e}^*, \mathrm{g}_{\mathrm{t}_e} \not\models \mathbf{Does}(\theta^d, v^t) \supset \mathbf{At}(v^t, \varphi)$, where $\mathrm{e}^* = \langle \mathrm{w}^*, \mathrm{a}, \mathrm{t} \rangle$ and $\mathrm{g}_{\mathrm{t}_e} = \{v^t \mapsto \mathrm{t}_e\}$. This means that $\mathrm{e}^*, \mathrm{g}_{\mathrm{t}_e} \models \mathbf{Does}(\theta^d, v^t)$ and $\mathrm{e}^*, \mathrm{g}_{\mathrm{t}_e} \not\models \mathbf{At}(v^t, \varphi)$. The latter means that $\langle \mathrm{w}^*, \mathrm{a}, \mathrm{t}_e \rangle, \mathrm{g}_{\mathrm{t}_e} \not\models \varphi$, which contradicts 1. ■

The same rule need not hold for **Res** since the validity of φ fails to ensure that it is possible to do the action next.

The next two propositions show that **AfterNec** and **Res** also distribute over conjunctions. The first proposition says that $\varphi_1 \wedge \varphi_2$ must hold after the agent does action θ^d if and only if φ_1 must hold after he does θ^d and φ_2 must hold after he does θ^d:

Proposition 3.66 $\models \mathbf{AfterNec}(\theta^d, \varphi_1 \wedge \varphi_2) \equiv \mathbf{AfterNec}(\theta^d, \varphi_1) \wedge \mathbf{AfterNec}(\theta^d, \varphi_2)$

The second proposition, which trivially follows from the first, says that the agent's doing action θ^d next must result in $\varphi_1 \wedge \varphi_2$ holding if and only if his doing θ^d next must result in φ_1 and his doing θ^d next must also result in φ_2:

Proposition 3.67 $\models \mathbf{Res}(\theta^d, \varphi_1 \wedge \varphi_2) \equiv \mathbf{Res}(\theta^d, \varphi_1) \wedge \mathbf{Res}(\theta^d, \varphi_2)$

We can also apply existential generalization inside **AfterNec** and **Res** operators. Thus, we have that if the agent's doing action θ^d next implies that φ must hold afterwards, then his doing it next implies that something is φ afterwards:

Proposition 3.68 $\models \mathbf{AfterNec}(\theta^d, \varphi) \supset \mathbf{AfterNec}(\theta^d, \exists v\varphi)$

Proof:
Clearly, $\models \varphi \supset \exists v\varphi$. Thus by proposition 3.65, we have that $\models \mathbf{AfterNec}(\theta^d, \varphi \supset \exists v\varphi)$. The result follows from this by proposition 3.63. ■

The analogous result for **Res** trivially follows: if the agent's doing θ^d next results in φ holding, then his doing it must also result in something being φ.

Proposition 3.69 $\models \mathbf{Res}(\theta^d, \varphi) \supset \mathbf{Res}(\theta^d, \exists v\varphi)$

The next proposition clarifies one aspect of **AfterNec** that may not be obvious. Remember that the empty action **skip** is the one that is done over a time interval if and only if the interval has a null duration; thus it does not require the agent to do anything and is always possible. Now as the following proposition shows, φ must hold after the agent does the empty action if and only if φ is necessary:

Proposition 3.70 $\models \mathbf{AfterNec}(\mathbf{skip}, \varphi) \equiv \Box\varphi$

This clearly implies that the mere fact that φ holds is insufficient for $\mathbf{AfterNec}(\mathbf{skip}, \varphi)$ to hold. Remember that φ may be contingent because it it may be making a claim about the future. And **AfterNec** requires its propositional argument to be necessary after the action has been done, even if the action takes no time.

The next result provides evidence that claims about states of affairs holding at an indexically specified time (e.g., **now**) receive proper temporal adjustment in the context of **AfterNec**. The proposition says that if φ necessarily holds **now**, then after the agent does θ^d next, it must be the case that he has just done θ^d from some time when φ held:

Proposition 3.71 $\models \Box\varphi \supset \mathbf{AfterNec}(\theta^d, \mathbf{DoneWhen}(\theta^d, \varphi))$

Proof:
1. Assume that $\mathsf{e}, \mathsf{g} \models \Box\varphi$.
2. Assume that $\mathsf{e}, \mathsf{g} \not\models \mathbf{AfterNec}(\theta^d, \mathbf{DoneWhen}(\theta^d, \varphi))$.
3. Let $\mathsf{e} = \langle \mathsf{w}, \mathsf{a}, \mathsf{t} \rangle$. Step 2 means that there exists w^*, and t_e such that $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$,

$$\langle \mathsf{w}^*, \mathsf{a}, \mathsf{t} \rangle, \mathsf{g}_e \models \mathbf{Does}(\theta^d, v^t_e) \qquad (3a)$$
and
$$\langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_e \rangle, \mathsf{g}_e \not\models \mathbf{DoneWhen}(\theta^d, \varphi) \qquad (3b)$$

where v^t_e is distinct from v^t and neither occurs free in φ, and where $\mathsf{g}_e = \{v^t_e \mapsto \mathsf{t}_e\}$.
4. Combining 3a with 1, we obtain that $\langle \mathsf{w}^*, \mathsf{a}, \mathsf{t} \rangle, \mathsf{g}_e \models \mathbf{Does}(\theta^d, v^t_e) \wedge \varphi$, which contradicts 3b. ■

The consequent does not hold if we just assume that φ holds. Proposition 3.70 should help in understanding why: if φ is makes a claim about the future and is merely contingent, it need not have been true in all $\approx_{\mathsf{t}}$-accessible worlds where the action has been done.

The next proposition looks at the same issue of temporal projection, but from the past to the present. It says that if action θ^d was done from some past time v^t to **now** and if at v^t, it was necessary that φ hold after θ^d was done, then φ necessarily holds now.

Proposition 3.72 $\models \mathbf{DoneWhen}(\theta^d, \mathbf{AfterNec}(\theta^d, \varphi)) \supset \Box\varphi$

Proof:
1. Assume that $\langle \mathsf{w}, \mathsf{a}, \mathsf{t} \rangle, \mathsf{g} \models \mathbf{DoneWhen}(\theta^d, \mathbf{AfterNec}(\theta^d, \varphi))$. This means that there exists t_s such that $\Delta([\![\theta^d]\!]_{\langle \mathsf{w}, \mathsf{a}, \mathsf{t}_s \rangle, \mathsf{g}}, \langle \mathsf{w}, \mathsf{a}, \mathsf{t}_s \rangle, \mathsf{t})$ and for all w^* and t_e, such that $\mathsf{w} \approx_{\mathsf{t}_s} \mathsf{w}^*$, either it is not the case that $\Delta([\![\theta^d]\!]_{\langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_s \rangle, \mathsf{g}}, \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_s \rangle, \mathsf{t})$, or $\langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_e \rangle, \mathsf{g} \models \varphi$ (we can

ignore the effects of expanding **DoneFrom** and **AfterNec** on g, because the variables introduced can't occur free in φ and θ^d).
2 Suppose that $\langle w, a, t\rangle, g \not\models \Box\varphi$, that is, that there exists a w_c, such that $w \approx_t w_c$ and $\langle w_c, a, t\rangle, g \not\models \varphi$.
3. By 1 and constraint 3.3, we must have that $t_s \preceq t$.
4. By this, 2, and constraint 3.5, we must have that $w \approx_{t_s} w_c$.
5. By this, 2, and 1, we must have that it is not the case that $\Delta([\![\theta^d]\!]_{\langle w_c,a,t_s\rangle,g}, \langle w_c, a, t_s\rangle, t)$.
6. By 4 and lemma 3.4, we must have that $[\![\theta^d]\!]_{\langle w_c,a,t_s\rangle,g} = [\![\theta^d]\!]_{\langle w,a,t_s\rangle,g}$. Thus by 5, we must have that $\Delta([\![\theta^d]\!]_{\langle w,a,t_s\rangle,g}, \langle w_c, a, t_s\rangle, t)$.
7. By this, 4 and constraint 3.6, we must then have that $\Delta([\![\theta^d]\!]_{\langle w,a,t_s\rangle,g}, \langle w, a, t_s\rangle, t)$. This contradicts 1. ■

Our final result shows that **AfterNec** applied to sequences of simple actions has the expected meaning. It says that φ must hold after θ^d_1 and θ^d_2 are done in sequence if and only if after doing θ^d_1, it must be the case that after doing θ^d_2, φ must hold.

Proposition 3.73 $\models \mathbf{AfterNec}((\theta^d_1;\theta^d_2),\varphi) \equiv \mathbf{AfterNec}(\theta^d_1, \mathbf{AfterNec}(\theta^d_2,\varphi))$

Proof:

1. $\langle w, a, t\rangle, g \models \mathbf{AfterNec}((\theta^d_1;\theta^d_2),\varphi)$

iff $\langle w, a, t\rangle, g \models \Box\forall v^t_2(\mathbf{Does}((\theta^d_1;\theta^d_2), v^t_2) \supset \mathbf{At}(v^t_2,\varphi))$, where v^t_2 doesn't occur free in φ

iff $\langle w, a, t\rangle, g \models \Box\forall v^t_2(\exists v^t_1(\mathbf{Does}(\theta^d_1, v^t_1) \wedge \mathbf{At}(v^t_1, \mathbf{Does}(\theta^d_2, v^t_2))) \supset \mathbf{At}(v^t_2,\varphi))$, where v^t_1 and v^t_2 are distinct and don't occur free in φ

iff for all w^* such that $w \approx_t w^*$ and for all t_2, either it is not the case that there exists a t_1 such that $\Delta([\![\theta^d_1]\!]_{\langle w^*,a,t\rangle,g}, \langle w^*, a, t\rangle, t_1)$ and $\Delta([\![\theta^d_2]\!]_{\langle w^*,a,t_1\rangle,g}, \langle w^*, a, t_1\rangle, t_2)$ or $\langle w^*, a, t_2\rangle, g \models \varphi$

iff for all w^* such that $w \approx_t w^*$ and for all t_1 and t_2, either it is not the case that $\Delta([\![\theta^d_1]\!]_{\langle w^*,a,t\rangle,g}, \langle w^*, a, t\rangle, t_1)$ or it is not the case that $\Delta([\![\theta^d_2]\!]_{\langle w^*,a,t_1\rangle,g}, \langle w^*, a, t_1\rangle, t_2)$ or $\langle w^*, a, t_2\rangle, g \models \varphi$

2. $\langle w, a, t\rangle, g \models \mathbf{AfterNec}(\theta^d_1, \mathbf{AfterNec}(\theta^d_2,\varphi))$

iff $\langle w, a, t\rangle, g \models \Box\forall v^t_1(\mathbf{Does}(\theta^d_1, v^t_1) \supset \mathbf{At}(v^t_1, \Box\forall v^t_2(\mathbf{Does}(\theta^d_2, v^t_2) \supset \mathbf{At}(v^t_2,\varphi))))$, where v^t_1 and v^t_2 are distinct and don't occur free in φ

iff for all w^* such that $w \approx_t w^*$ and for all t_1, if $\Delta([\![\theta^d_1]\!]_{\langle w^*,a,t\rangle,g}, \langle w^*, a, t\rangle, t_1)$ then for all w^{**} such that $w^* \approx_{t_1} w^{**}$ and for all t_2, if $\Delta([\![\theta^d_2]\!]_{\langle w^{**},a,t_1\rangle,g}, \langle w^{**}, a, t_1\rangle, t_2)$, then $\langle w^{**}, a, t_2\rangle, g \models \varphi$

3. Since by constraint 3.4, $\approx_{t_1}$ must be reflexive, 2 implies that for all w^* such that $w \approx_t w^*$ and for all t_1, if $\Delta([\![\theta^d_1]\!]_{\langle w^*,a,t\rangle,g}, \langle w^*, a, t\rangle, t_1)$ then for all t_2, if $\Delta([\![\theta^d_2]\!]_{\langle w^*,a,t_1\rangle,g}, \langle w^*, a, t_1\rangle, t_2)$, then $\langle w^*, a, t_2\rangle, g \models \varphi$, which is clearly equivalent to 1.
4. Now let's show that 1 implies 2. Suppose that 1 holds and that 2 does not. The latter means that there exists a w^* and a t_1, such that $w \approx_t w^*$, $\Delta([\![\theta^d_1]\!]_{\langle w^*,a,t\rangle,g}, \langle w^*, a, t\rangle, t_1)$,

and there exists a w^{**} and a t_2, such that $w* \approx_{t_1} w^{**}$, $\Delta([\![\theta_2^d]\!]_{\langle w^{**},a,t_1\rangle,g}, \langle w^{**}, a, t_1\rangle, t_2)$, and $\langle w^{**}, a, t_2\rangle, g \not\models \varphi$. Since $\Delta([\![\theta_1^d]\!]_{\langle w^*,a,t\rangle,g}, \langle w^*, a, t\rangle, t_1)$, by constraint 3.3, we must have that $t \preceq t_1$. By constraint 3.5, this and $w* \approx_{t_1} w^{**}$ imply that $w* \approx_t w^{**}$. Thus by lemma 3.4, we must have that $[\![\theta_1^d]\!]_{\langle w^*,a,t\rangle,g} = [\![\theta_1^d]\!]_{\langle w^{**},a,t\rangle,g}$. Since $w* \approx_{t_1} w^{**}$ and $\Delta([\![\theta_1^d]\!]_{\langle w^*,a,t\rangle,g}, \langle w^*, a, t\rangle, t_1)$, by the above and constraint 3.6, we must then have that $\Delta([\![\theta_1^d]\!]_{\langle w^{**},a,t\rangle,g}, \langle w^{**}, a, t\rangle, t_1)$. Finally, since $w \approx_t w^*$ and $w* \approx_t w^{**}$, by constraint 3.4, we must have that $w \approx_t w^{**}$. So it must be the case that there exists w^{**}, t_1, and t_2, such that $w \approx_t w^{**}$, $\Delta([\![\theta_1^d]\!]_{\langle w^*,a,t\rangle,g}, \langle w^{**}, a, t\rangle, t_1)$, $\Delta([\![\theta_2^d]\!]_{\langle w^{**},a,t_1\rangle,g}, \langle w^{**}, a, t_1\rangle, t_2)$, and $\langle w^{**}, a, t_2\rangle, g \not\models \varphi$. This contradicts 1, so 1 must imply 2. ∎

3.5.7 The Dynamics of Knowledge

Now let's move on to the issue of how what is known changes over time. In section 3.1, we argued that in order to be able to show in a reasonably simple way that agents can achieve goals by doing multi-step actions, we should assume that knowledge is persistent, that is, that agents keep on knowing what they knew previously, and that agents know what primitive actions they have just done. We will now discuss the exact form that this assumption will take and examine some of its consequences.

If we ignore indexical knowledge, it is easy to formalize the requirement that knowledge be persistent. In this case, the knowledge states of a set of agents over time can be modeled by an indexed family of accessibility relations over possible worlds — one relation for each combination of agent and time. To ensure that knowledge is persistent, we need to require something like the following: that once a world has been ruled out by an agent, it can never be ruled in again later. More technically then, knowledge will be persistent if we require that if $\langle w, w'\rangle \in K_{a,t}$ and $t_p \preceq t$, then $\langle w, w'\rangle \in K_{a,t_p}$.

When we start taking indexical knowledge into account, this must be adapted. Temporally indexical knowledge must be adjusted for the passage of time; for example, John's knowing that it is now raining must later become something like John's knowing that it was raining earlier. We are now modeling knowledge using a relation over indices. If we simply replace "K-accessible world" by "K-accessible index" in the above constraint, we get the following requirement: if $\langle\langle w, a, t\rangle, e'\rangle \in K$ and $t_p \preceq t$, then $\langle\langle w, a, t_p\rangle, e'\rangle \in K$. But this is incorrect! The intuition behind K-accessible indices is that they model the world plus point-of-view combinations that are not ruled out by what the agent knows at the time. But to say that if an index is K-accessible at one time, it must be K-accessible at any earlier time, implies that agents always think that time stands still. It must be possible to adjust the time component of the index to take into account the passage of time. But on the other hand, we do not want to go as far as requiring agents to know exactly how much time has elapsed since some previous state.

Additionally, we want agents to be aware of what primitive actions they have just done. And we also want them to remember the association between the temporally indexical knowledge they had and the action they have done: if they knew that some state of affairs was holding at the time they started doing the action, then they should know after the action that that state of affairs was holding at whatever time they started doing it.

The following constraint on semantic structures, reproduced from section 3.3.3, captures all these requirements:

Constraint 3.8
If $\langle\langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle, \langle \mathsf{w}', \mathsf{a}', \mathsf{t}'\rangle\rangle \in \mathsf{K}$ and $\mathsf{t}_p \preceq \mathsf{t}$, then there exists a time t'_p, where $\mathsf{t}'_p \preceq \mathsf{t}'$, such that $\langle\langle \mathsf{w}, \mathsf{a}, \mathsf{t}_p\rangle, \langle \mathsf{w}', \mathsf{a}', \mathsf{t}'_p\rangle\rangle \in \mathsf{K}$ and if $\Delta(\mathsf{d}, \langle \mathsf{w}, \mathsf{a}, \mathsf{t}_p\rangle, \mathsf{t})$ then $\Delta(\mathsf{d}, \langle \mathsf{w}', \mathsf{a}', \mathsf{t}'_p\rangle, \mathsf{t}')$.

Let's look at what this is saying. If we ignore for the moment the last part concerning Δ, this says essentially that if an index is ruled in at a given time, then that index, with an appropriate temporal adjustment, must have been ruled in at any earlier time. The adjustment involves a change to the temporal component of the index so that it is earlier or equal to what it originally was. More formally, the constraint says that if an index $\langle \mathsf{w}', \mathsf{a}', \mathsf{t}'\rangle$ is compatible with what an agent a knows at a time t in world w, then an adjusted version of that index $\langle \mathsf{w}', \mathsf{a}', \mathsf{t}'_p\rangle$ must have been compatible with what a knew at an earlier time t_p in world w, where this adjusted version must be identical to the original index in all but its temporal component, and where this temporal component t'_p must be prior to or equal to the temporal component t' of the original index.

The following proposition shows that the constraint effectively requires knowledge to be persistent and that temporally indexical knowledge is properly adjusted. The proposition can be liberally paraphrased as saying that if an agent knows that φ, then he will later know that it used to be the case that φ.

Proposition 3.74 (Persistence of Knowledge)

$$\models \mathbf{Know}(\varphi) \supset \mathbf{AllFutN}(\mathbf{Know}(\mathbf{SomePastN}(\varphi)))$$

Proof:
1. Assume that $\langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle, \mathsf{g} \models \mathbf{Know}(\varphi)$.
2. Take an arbitrary time t_f, such that $\mathsf{t} \preceq \mathsf{t}_f$, and take an arbitrary index $\langle \mathsf{w}', \mathsf{a}', \mathsf{t}'_f\rangle$, such that $\langle\langle \mathsf{w}, \mathsf{a}, \mathsf{t}_f\rangle, \langle \mathsf{w}', \mathsf{a}', \mathsf{t}'_f\rangle\rangle \in \mathsf{K}$. By constraint 3.8, it must be the case that there exists t' such that $\mathsf{t}' \preceq \mathsf{t}'_f$ and $\langle\langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle, \langle \mathsf{w}', \mathsf{a}', \mathsf{t}'\rangle\rangle \in \mathsf{K}$.
3. By 1 and 2, it must then be the case that $\langle \mathsf{w}', \mathsf{a}', \mathsf{t}'\rangle, \mathsf{g} \models \varphi$. Since $\mathsf{t}' \preceq \mathsf{t}'_f$, we must also have that $\langle \mathsf{w}', \mathsf{a}', \mathsf{t}'_f\rangle, \mathsf{g} \models \mathbf{SomePastN}(\varphi)$.
4. Since $\langle \mathsf{w}', \mathsf{a}', \mathsf{t}'_f\rangle$ is an arbitrary K-alternative to $\langle \mathsf{w}, \mathsf{a}, \mathsf{t}_f\rangle$, 3 implies that $\langle \mathsf{w}, \mathsf{a}, \mathsf{t}_f\rangle, \mathsf{g} \models \mathbf{Know}(\mathbf{SomePastN}(\varphi))$.
5. Finally since t_f is an arbitrary time such that $\mathsf{t} \preceq \mathsf{t}_f$, 4 implies that $\langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle, \mathsf{g} \models \mathbf{AllFutN}(\mathbf{Know}(\mathbf{SomePastN}(\varphi)))$. ■

Now let's look at the part of the constraint involving Δ. It requires that if the agent a has actually done some primitive action d from time t_p to time t, then the agent a′ of the K-accessible index (the one a thinks he might be) must have done d from time t'_p (the time a used to think might be the current time) to time t′ (the time a now thinks might be the current time) in the world w′ of the K-accessible index. Thus, the agent must know at time t that he has just done primitive action d. The following proposition verifies this:

Proposition 3.75 (Awareness of Own Primitive Actions)

$$\models \forall d\, \mathbf{AfterNec}(d, \mathbf{Know}(\mathbf{Done}(d)))$$

Proof:
1. Suppose that $\not\models \forall d\, \mathbf{AfterNec}(d, \mathbf{Know}(\mathbf{Done}(d)))$.
$\not\models \forall d\, \mathbf{AfterNec}(d, \mathbf{Know}(\mathbf{Done}(d)))$
iff there exists w, a, t, g, and d, such that
$\langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle, \mathsf{g}\{d \mapsto \mathsf{d}\} \not\models \mathbf{AfterNec}(d, \mathbf{Know}(\mathbf{Done}(d)))$
iff there exists w, a, t, g, d, w_*, and t_e such that
$\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}_*$, $\Delta(\mathsf{d}, \langle \mathsf{w}_*, \mathsf{a}, \mathsf{t}\rangle, \mathsf{t}_e)$, and $\langle \mathsf{w}_*, \mathsf{a}, \mathsf{t}_e\rangle, \mathsf{g}\{d \mapsto \mathsf{d}\} \not\models \mathbf{Know}(\mathbf{Done}(d))$
iff there exists w, a, t, g, d, w_*, and t_e such that $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}_*$, $\Delta(\mathsf{d}, \langle \mathsf{w}_*, \mathsf{a}, \mathsf{t}\rangle, \mathsf{t}_e)$, and
there exists w'_*, a′, and t'_e such that $\langle\langle \mathsf{w}_*, \mathsf{a}, \mathsf{t}_e\rangle, \langle \mathsf{w}'_*, \mathsf{a}', \mathsf{t}'_e\rangle\rangle \in \mathsf{K}$
and $\langle \mathsf{w}'_*, \mathsf{a}', \mathsf{t}'_e\rangle, \mathsf{g}\{d \mapsto \mathsf{d}\} \not\models \mathbf{Done}(d)$
iff there exists w, a, t, g, d, w_*, and t_e such that $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}_*$, $\Delta(\mathsf{d}, \langle \mathsf{w}_*, \mathsf{a}, \mathsf{t}\rangle, \mathsf{t}_e)$, and
there exists w'_*, a′, and t'_e such that $\langle\langle \mathsf{w}_*, \mathsf{a}, \mathsf{t}_e\rangle, \langle \mathsf{w}'_*, \mathsf{a}', \mathsf{t}'_e\rangle\rangle \in \mathsf{K}$ and
for all t_s such that $\mathsf{t}_s \preceq \mathsf{t}'_e$, it is not the case that $\Delta(\mathsf{d}, \langle \mathsf{w}'_*, \mathsf{a}', \mathsf{t}_s\rangle, \mathsf{t}'_e)$
2. Notice that we have that $\langle\langle \mathsf{w}_*, \mathsf{a}, \mathsf{t}_e\rangle, \langle \mathsf{w}'_*, \mathsf{a}', \mathsf{t}'_e\rangle\rangle \in \mathsf{K}$, $\mathsf{t} \preceq \mathsf{t}_e$, and $\Delta(\mathsf{d}, \langle \mathsf{w}_*, \mathsf{a}, \mathsf{t}\rangle, \mathsf{t}_e)$. So by constraint 3.8, we have that there exists t′ such that $\mathsf{t}' \preceq \mathsf{t}$ and $\Delta(\mathsf{d}, \langle \mathsf{w}'_*, \mathsf{a}', \mathsf{t}'\rangle, \mathsf{t}'_e)$. This contradicts 1. ■

But note that this only applies to primitive actions, not to arbitrary action description. Thus, $\mathbf{AfterNec}(\theta^d, \mathbf{Know}(\mathbf{Done}(\theta^d)))$ is not valid. This is because the agent may not know that θ^d denotes the primitive action he has just done. For example, I may be aware that I have just done the action of "turning right" without realizing that in virtue of that I have also done the action of "turning towards Kingston", which is in the opposite direction from where I want to go.

Yet, there is still more to our constraint. The agent must also know that the time at which he started doing the action d is just the time that he used to think of as the current time when he started doing d, and therefore that whatever was then known to be currently true is now known to be true at whatever time he started doing the just completed action d. The following proposition shows that the constraint ensures that agents know what action they have just done and that if they knew that some state of affairs was holding when they started doing the action, they now know that that state of affairs was holding at whatever time they started doing the action:[20]

Proposition 3.76 $\models \forall d(\mathbf{Know}(\varphi) \supset \mathbf{AfterNec}(d, \mathbf{Know}(\mathbf{DoneWhen}(d,\varphi))))$

Proof:

1. Suppose that

$$\not\models \forall d(\mathbf{Know}(\varphi) \supset \mathbf{AfterNec}(d, \mathbf{Know}(\mathbf{DoneWhen}(d,\varphi))))$$

Now by the definition of satisfaction,

$\not\models \forall d(\mathbf{Know}(\varphi) \supset \mathbf{AfterNec}(d, \mathbf{Know}(\mathbf{DoneWhen}(d,\varphi))))$
iff there exists w, a, t, g, and d, such that $\langle w,a,t\rangle, g\{d \mapsto d\} \models \mathbf{Know}(\varphi)$ and
$\langle w,a,t\rangle, g\{d \mapsto d\} \not\models \mathbf{AfterNec}(d, \mathbf{Know}(\mathbf{DoneWhen}(d,\varphi)))$
iff there exists w, a, t, g, and d, such that $\langle w,a,t\rangle, g\{d \mapsto d\} \models \mathbf{Know}(\varphi)$
and there exists w_*, and t_e such that $w \approx_t w_*$, $\Delta(d, \langle w_*, a, t\rangle, t_e)$,
and $\langle w_*, a, t_e\rangle, g\{d \mapsto d\} \not\models \mathbf{Know}(\mathbf{DoneWhen}(d,\varphi))$
iff there exists w, a, t, g, and d, such that $\langle w,a,t\rangle, g\{d \mapsto d\} \models \mathbf{Know}(\varphi)$
and there exists w_*, and t_e such that $w \approx_t w_*$, $\Delta(d, \langle w_*, a, t\rangle, t_e)$, and
there exists w'_*, a', and t'_e such that $\langle\langle w_*, a, t_e\rangle, \langle w'_*, a', t'_e\rangle\rangle \in K$
and $\langle w'_*, a', t'_e\rangle, g\{d \mapsto d\} \not\models \mathbf{DoneWhen}(d,\varphi)$
iff there exists w, a, t, g, and d, such that $\langle w,a,t\rangle, g\{d \mapsto d\} \models \mathbf{Know}(\varphi)$ and
there exists w_*, and t_e such that $w \approx_t w_*$, $\Delta(d, \langle w_*, a, t\rangle, t_e)$, and
there exists w'_*, a', and t'_e such that $\langle\langle w_*, a, t_e\rangle, \langle w'_*, a', t'_e\rangle\rangle \in K$ and
for all t_s such that $t_s \preceq t'_e$, either it is not the case
that $\Delta(d, \langle w'_*, a', t_s\rangle, t'_e)$, or $\langle w'_*, a', t_s\rangle, g\{d \mapsto d\} \not\models \varphi$

2. Notice that we have that $\langle\langle w_*, a, t_e\rangle, \langle w'_*, a', t'_e\rangle\rangle \in K$, $t \preceq t_e$, and $\Delta(d, \langle w_*, a, t\rangle, t_e)$. So by constraint 3.8, we have that there exists t' such that $t' \preceq t$, $\langle\langle w_*, a, t\rangle, \langle w'_*, a', t'\rangle\rangle \in K$, and $\Delta(d, \langle w'_*, a', t'\rangle, t'_e)$.

[20] Given what our constraint says, it should be possible to prove an analogous result for cases where the agent does several actions concurrently.

3. We have that $\langle w, a, t\rangle, g\{d \mapsto d\} \models \mathbf{Know}(\varphi)$, so by proposition 3.46, we must also have that $\langle w, a, t\rangle, g\{d \mapsto d\} \models \Box\mathbf{Know}(\varphi)$.
4. Since by 1, $w \approx_t w_*$ and by 2, $\langle\langle w_*, a, t\rangle, \langle w'_*, a', t'\rangle\rangle \in K$, 3 means that it must be the case that $\langle w'_*, a', t_s\rangle, g\{d \mapsto d\} \models \varphi$. This, together with 2 means that there exists t' such that $t' \preceq t$, $\Delta(d, \langle w'_*, a', t'\rangle, t'_e)$, and $\langle w'_*, a', t_s\rangle, g\{d \mapsto d\} \models \varphi$. This contradicts 1. ■

The above result can be "chained" to show that after doing a sequence of primitive actions, the agent knows that he has just done that sequence of actions, and whatever states of affairs were then known to hold are now known to have been holding at whatever time he started doing the actions. Thus, we get the following proposition:

Proposition 3.77

$$\models \forall d_1 \forall d_2(\mathbf{Know}(\varphi) \supset \mathbf{AfterNec}((d_1; d_2), \mathbf{Know}(\mathbf{DoneWhen}((d_1; d_2), \varphi))))$$

Proof:
1. Assume that $e, g \models \mathbf{Know}(\varphi)$.
2. By propositions 3.76 and 3.3, this implies that

$$e, g \models \mathbf{AfterNec}(d_1, \mathbf{Know}(\mathbf{DoneWhen}(d_1, \varphi)))$$

3. By propositions 3.76, 3.3, and 3.65, we have that

$$\models \mathbf{AfterNec}(d_1, \mathbf{Know}(\varphi') \supset \mathbf{AfterNec}(d_2, \mathbf{Know}(\mathbf{DoneWhen}(d_2, \varphi'))))$$

And by proposition 3.63, this yields that

$$\begin{aligned}&\models \mathbf{AfterNec}(d_1, \mathbf{Know}(\varphi')) \supset\\ &\quad \mathbf{AfterNec}(d_1, \mathbf{AfterNec}(d_2, \mathbf{Know}(\mathbf{DoneWhen}(d_2, \varphi'))))\end{aligned}$$

4. So by 2 and 3, we get that

$$\begin{aligned}e, g \models\ & \mathbf{AfterNec}(d_1, \mathbf{AfterNec}(d_2, \mathbf{Know}(\\ &\qquad \mathbf{DoneWhen}(d_2, \mathbf{Know}(\mathbf{DoneWhen}(d_1, \varphi))))))\end{aligned}$$

5. Let us show that

$$\models \mathbf{DoneWhen}(d_2, \mathbf{Know}(\mathbf{DoneWhen}(d_1, \varphi))) \supset \mathbf{DoneWhen}((d_1; d_2), \varphi)$$

(a) Assume that

$$\langle w, a, t\rangle, g \models \mathbf{DoneWhen}(d_2, \mathbf{Know}(\mathbf{DoneWhen}(d_1, \varphi)))$$

This means that there exists t_i such that $\Delta(g(d_2), \langle w, a, t_i\rangle, t)$ and

$$\langle w, a, t_i\rangle, g \models \mathbf{Know}(\mathbf{DoneWhen}(d_1, \varphi))$$

(b) By proposition 3.12, this implies that there exists t_i such that $\Delta(g(d_2), \langle w, a, t_i\rangle, t)$ and there exists t_s such that $\Delta(g(d_1), \langle w, a, t_s\rangle, t_i)$ and $\langle w, a, t_s\rangle, g \models \varphi$.

(c) This holds

iff there exists t_s such that $\Delta(\mathrm{g}(d_1), \langle \mathrm{w}, \mathrm{a}, \mathrm{t}_s \rangle, \mathrm{t}_i)$ and there exists t_i such that
$\Delta(\mathrm{g}(d_2), \langle \mathrm{w}, \mathrm{a}, \mathrm{t}_i \rangle, \mathrm{t})$ and $\langle \mathrm{w}, \mathrm{a}, \mathrm{t}_s \rangle, \mathrm{g} \models \varphi$

iff $\langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g} \models$
$\exists v^t(\exists v^t_e(v^t_e = \mathbf{now} \land \mathbf{At}(v^t, \exists v^t_i(\mathbf{Does}(d_1, v^t_i) \land \mathbf{At}(v^t_i, \mathbf{Does}(d_2, v^t_e))))) \land \mathbf{At}(v^t, \varphi))$

iff $\langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g} \models \exists v^t(\exists v^t_e(v^t_e = \mathbf{now} \land \mathbf{At}(v^t, \mathbf{Does}((d_1; d_2), v^t_e))) \land \mathbf{At}(v^t, \varphi))$

iff $\langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g} \models \exists v^t(\mathbf{DoneFrom}((d_1; d_2), v^t) \land \mathbf{At}(v^t, \varphi))$

iff $\langle \mathrm{w}, \mathrm{a}, \mathrm{t} \rangle, \mathrm{g} \models \mathbf{DoneWhen}((d_1; d_2), \varphi)$

6. By proposition 3.17 and 3.11, we have that

$$\text{if} \models \varphi_1 \supset \varphi_2, \text{ then} \models \mathbf{Know}(\varphi_1) \supset \mathbf{Know}(\varphi_2)$$

Then by proposition 3.65 and 3.63, we have that

$$\text{if} \models \varphi_1 \supset \varphi_2, \text{ then} \models \mathbf{AfterNec}(d_2, \mathbf{Know}(\varphi_1)) \supset \mathbf{AfterNec}(d_2, \mathbf{Know}(\varphi_2))$$

And by the same propositions, we finally get that

if $\models \varphi_1 \supset \varphi_2$, then
$\models \mathbf{AfterNec}(d_1, \mathbf{AfterNec}(d_2, \mathbf{Know}(\varphi_1))) \supset \mathbf{AfterNec}(d_1, \mathbf{AfterNec}(d_2, \mathbf{Know}(\varphi_2)))$

Then, this together with 4 and 5 implies that

$$\mathrm{e}, \mathrm{g} \models \mathbf{AfterNec}(d_1, \mathbf{AfterNec}(d_2, \mathbf{Know}(\mathbf{DoneWhen}((d_1; d_2), \varphi))))$$

7. By proposition 3.73, this implies that

$$\mathrm{e}, \mathrm{g} \models \mathbf{AfterNec}((d_1; d_2), \mathbf{Know}(\mathbf{DoneWhen}((d_1; d_2), \varphi)))$$

■

This implies that agents effectively remember the entire history of their own primitive actions and retain associations between the various pieces of temporally indexical knowledge they have with particular points (or intervals) in this history. An agent's knowledge of the history of his primitive actions provides him with something akin to an internal clock, which he can use to "time stamp" his indexical knowledge.

When reasoning about the ability of an agent to achieve goals by doing actions, sequences of actions in particular, it is not so much in the agent's remembering of what states of affairs held when he started that we are interested, but in the following: if he knows that some state of affairs, typically an intermediate state on the way to a goal state, would result from his doing an action, then his doing it would result in his knowing that that state is in fact holding; this would ensure that he would have enough knowledge to be able achieve the goal state by doing the remaining actions in the sequence. The next two propositions show that, as a result of constraint 3.8, knowledge does in fact have this property:

Proposition 3.78 $\models \forall d(\mathbf{Know}(\mathbf{AfterNec}(d,\varphi)) \supset \mathbf{AfterNec}(d,\mathbf{Know}(\varphi)))$

Proof:
1. By proposition 3.76, we have that

$$\models \mathbf{Know}(\mathbf{AfterNec}(d,\varphi)) \supset \mathbf{AfterNec}(d,\mathbf{Know}(\mathbf{DoneWhen}(d,\mathbf{AfterNec}(d,\varphi))))$$

2. By propositions 3.72 and 3.38, we have that

$$\models \mathbf{DoneWhen}(d,\mathbf{AfterNec}(d,\varphi)) \supset \varphi$$

So by propositions 3.17 and 3.11, it must be the case that

$$\models \mathbf{Know}(\mathbf{DoneWhen}(d,\mathbf{AfterNec}(d,\varphi))) \supset \mathbf{Know}(\varphi)$$

Then by propositions 3.65 and 3.63, we must also have that

$$\models \mathbf{AfterNec}(d,\mathbf{Know}(\mathbf{DoneWhen}(d,\mathbf{AfterNec}(d,\varphi)))) \supset \mathbf{AfterNec}(d,\mathbf{Know}(\varphi))$$

This combines with 1 to yield the desired result. ■

It follows trivially that:

Proposition 3.79 $\models \forall d(\mathbf{Know}(\mathbf{Res}(d,\varphi)) \supset \mathbf{Res}(d,\mathbf{Know}(\varphi)))$

3.6 A Formalization of Ability

In this section, we describe our formalization of ability. Its advantages and shortcomings are discussed in the next section. Our primary objective in developing this formalization is not to obtain an ability operator that will correspond perfectly to the intuitive notion or that will handle any kind of goal or action. Rather, it is to show that our account of indexical knowledge can be exploited in order to make existing formalizations of ability less demanding in terms of the knowledge that they require. The kind of situations where this yields significant benefits are cases where mere indexical knowledge is sufficient for ability to achieve a goal and *de re* knowledge is unnecessary. For example, even if I am lost in a foreign city, if I see a phone booth across the street, I am still able to phone my hotel and find out how to get back there. Even if I have been working so hard that I lost track of time, I can still get up and go make some coffee. Even if I drive off the road, bump my head, and forget who I am, I can still wave to a passing car to get some help. Existing frameworks for talking about ability cannot handle these cases properly as they do not accommodate the distinction between indexical knowledge and *de re* knowledge. Thus, they tend to require *de re* knowledge when mere indexical knowledge would be sufficient for ability. Because our framework does capture the distinction, we can reflect more faithfully the true requirements of such situations.

We base our formalization of ability on that of Moore (1980), which in spite of its relative simplicity, does get at the essential connection between the ability of agents to achieve goals and the knowledge they have about relevant actions. It is simpler than his because we do not attempt to handle indefinite iteration (while-loop actions). As discussed in section 2.1.1, Moore's formalization of this case is actually defective because it does not require the agent to know that the action will eventually terminate. We leave this case for future research.

Since we are not treating indefinite iteration, we can simply define ability in terms of the other constructs of the logic as follows:

Definition 3.8

$\mathbf{Can}(\theta^d, \varphi) \overset{\text{def}}{=} \exists v^d\, \mathbf{Know}(v^d = \theta^d \wedge \mathbf{Res}(\theta^d, \varphi))$, where action variable v^d does not occur free in φ and θ^d

$\mathbf{Can}(\mathbf{skip}, \varphi) \overset{\text{def}}{=} \mathbf{Know}(\varphi)$

$\mathbf{Can}((\delta_1; \delta_2), \varphi) \overset{\text{def}}{=} \mathbf{Can}(\delta_1, \mathbf{Can}(\delta_2, \varphi))$

$\mathbf{Can}(\mathbf{if}(\varphi_c, \delta_1, \delta_2), \varphi_g) \overset{\text{def}}{=} (\mathbf{Know}(\varphi_c) \wedge \mathbf{Can}(\delta_1, \varphi_g)) \vee (\mathbf{Know}(\neg\varphi_c) \wedge \mathbf{Can}(\delta_2, \varphi_g))$

The various cases of the definition are in most respects straight adaptations of Moore's axioms into our framework. As mentioned earlier, their point is to make an agent's ability to achieve various goals dependent upon his having the right kind of knowledge. What knowledge an agent must have in order to be able to achieve a goal by doing an action depends on the structure of that action. The definition works by recursion on the structure of the action expressions involved.

The first case takes care of simple actions — actions that are represented by action terms. In this case, **self** is said to be able to achieve a goal φ by doing a simple action θ^d if and only if he knows what that action is and knows that his doing it results in the goal holding afterwards.[21] Note that the definition involves quantifying-in over the class of primitive actions (e.g. "send grasping signal to hand"), as opposed to arbitrary action descriptions. The second case handles the empty action **skip**: **self** is able to achieve a goal by doing **skip** if and only if he knows that the goal currently holds. The third assumption handles sequentially composed actions: **self** is able to achieve a goal φ by doing $(\delta_1; \delta_2)$ if and only if by doing δ_1, he is able to achieve the goal that consists in himself being able to achieve the original goal φ by doing action δ_2. The final case takes care of conditional actions: **self** can achieve a goal by doing $\mathbf{if}(\varphi_c, \delta_1, \delta_2)$ if and only if he either knows that the condition φ_c holds and is able to achieve the goal by doing δ_1, or knows that it does not hold and is able to achieve the goal by doing δ_2.

Note that we eliminate Moore's requirement that the agent know who he is; instead, we require indexical knowledge. Thus, in the simple action case we require that the agent know what the action is, that he know that it is physically possible for *himself* to do it, and that he know that if *he himself* does it, the goal will necessarily hold afterwards. Similarly, in the sequential action case, we require that by doing the first action, the agent be able to achieve the goal of *himself* being able to achieve the original goal φ by doing the second action. Mere *de re* knowledge is neither necessary nor sufficient for being able to achieve a goal; we discuss this further and give a concrete example in section 4.3.

Note also that the adequacy of this formalization depends on adopting a particular interpretation of ability assertions, in particular, of the role that the given action plays in the ability of the agent to achieve the goal: it requires the agent to be able to achieve the goal by executing in a dumb manner a program that corresponds to the action expression given. This manifests itself in the constraint that is imposed on ability assertions involving conditional actions; this constraint requires that the agent know whether the condition holds before doing either branch of the action, even though nothing requires the branches to involve distinct actions. A smart interpreter might be able to detect that the branches share an initial action sequence and thus delay the need to know whether the condition holds. One could also view the action given as a description in the eye of an external

[21]By proposition 3.62,

$$\models \exists v^d \,\mathrm{Know}(v^d = \theta^d \wedge \mathrm{Res}(\theta^d, \varphi)) \equiv \exists v^d \,\mathrm{Know}(v^d = \theta^d \wedge \mathrm{Res}(v^d, \varphi)),$$

where action variable v^d does not occur free in φ and θ^d

Although it might be more natural to use the formula on the right in our definition, we prefer to use the one on the left because it leads to simpler proofs of ability.

observer of what courses of action one should consider in judging whether the agent is able to achieve the goal, in which case, any action expression that specify the same courses of action should result in the same ability requirements (this is the view taken by Cohen and Levesque (1987)). Each of these views seems coherent and each seems to have its domain of relevance. It is not clear that the one adopted here is the most appropriate for the kind of theory we are developing, but it is more easily formalized, and this is why it was taken.

For sequentially composed actions, it is rather unwieldy to develop proofs of ability that directly establish the formula that the desired ability statement stands for. The following proposition provides a convenient rule for decomposing the task of proving such results:

Proposition 3.80 If $\models \varphi_i \supset \mathbf{Can}(\delta, \varphi_e)$, then $\models \mathbf{Can}(\theta^d, \varphi_i) \supset \mathbf{Can}((\theta^d; \delta), \varphi_e)$

Proof:
1. Assume that $\models \varphi_i \supset \mathbf{Can}(\delta, \varphi_e)$.
2. Assume also that $\mathsf{e}, \mathsf{g} \models \mathbf{Can}(\theta^d, \varphi_i)$. This means that there exists a d, such that for all $\mathsf{e}', \langle \mathsf{e}, \mathsf{e}' \rangle \in \mathsf{K}$,

$$\mathsf{e}', \mathsf{g_d} \models d = \theta^d \wedge \mathbf{PhyPoss}(\theta^d) \wedge \mathbf{AfterNec}(\theta^d, \varphi_i)$$

where $\mathsf{g_d} = \mathsf{g}\{d \mapsto \mathsf{d}\}$.
3. By 1 and proposition 3.65, we must have that $\models \mathbf{AfterNec}(\theta^d, \varphi_i \supset \mathbf{Can}(\delta, \varphi_e))$.
4. By proposition 3.63, we have that

$$\models \mathbf{AfterNec}(\theta^d, \varphi_i \supset \mathbf{Can}(\delta, \varphi_e)) \supset (\mathbf{AfterNec}(\theta^d, \varphi_i) \supset \mathbf{AfterNec}(\theta^d, \mathbf{Can}(\delta, \varphi_e)))$$

5. So by 2, 3, and 4, it must be the case that $\mathsf{e}', \mathsf{g_d} \models \mathbf{AfterNec}(\theta^d, \mathbf{Can}(\delta, \varphi_e))$. Putting this together with 2, we obtain that $\mathsf{e}, \mathsf{g} \models \mathbf{Can}(\theta^d, \mathbf{Can}(\delta, \varphi_e))$. ■

On the basis of this rule, one can prove in the following manner that the agent is able to achieve a goal φ_e by doing the action $(\theta^d; \delta)$: first show that the agent is able to achieve some intermediate goal φ_i by doing the first action θ^d, and then show that φ_i's holding is sufficient for the agent being able to achieve the main goal φ_e by doing the second action δ. Since ability to achieve the main goal by doing the second action requires knowledge, the intermediate goal should represent the appropriate knowledge requirements.

From a planning point of view, what is most interesting is perhaps not what agents are actually able to do, but what they know that they are able to do. The next proposition shows that if the agent is able to achieve a goal by doing an action, then he must know that he is:

Proposition 3.81 $\models \mathbf{Can}(\delta, \varphi) \supset \mathbf{Know}(\mathbf{Can}(\delta, \varphi))$

Proof:
We show this by induction on the number of action composition operators in δ.

Base case: (a) In case $\delta = \theta^d$. By proposition 3.14, we have that $\models \exists v\mathbf{Know}(\varphi) \supset \mathbf{Know}(\exists v\mathbf{Know}(\varphi))$. The desired results follows immediately.
(b) In case $\delta = \mathbf{skip}$. Then, $\mathbf{Can}(\delta, \varphi) \overset{\text{def}}{=} \mathbf{Know}(\varphi)$. The desired results follows immediately by proposition 3.14.
Induction step: Assume that the result holds for all subactions of δ. We will show that it must then also hold for δ.
(a) In case $\delta = \mathbf{if}(\varphi_c, \delta_1, \delta_2)$. Then

$$\mathbf{Can}(\delta, \varphi) \overset{\text{def}}{=} (\mathbf{Know}(\varphi_c) \wedge \mathbf{Can}(\delta_1, \varphi)) \vee (\mathbf{Know}(\neg\varphi_c) \wedge \mathbf{Can}(\delta_2, \varphi))$$

By proposition 3.14 and the induction hypothesis, we must have that

$$\begin{aligned}\models \mathbf{Can}(\delta, \varphi) \equiv\ & (\mathbf{Know}(\mathbf{Know}(\varphi_c)) \wedge \mathbf{Know}(\mathbf{Can}(\delta_1, \varphi))) \vee \\ & (\mathbf{Know}(\mathbf{Know}(\neg\varphi_c)) \wedge \mathbf{Know}(\mathbf{Can}(\delta_2, \varphi)))\end{aligned}$$

The desired result follows easily.
(b) In case $\delta = (\delta_1; \delta_2)$. Then $\mathbf{Can}(\delta, \varphi) \overset{\text{def}}{=} \mathbf{Can}(\delta_1, \mathbf{Can}(\delta_2, \varphi))$. The desired result follows immediately by the induction hypothesis. ∎

The converse is also valid (by proposition 3.12). So even if one is concerned with planning, nothing is lost by looking at what agents are actually able to do.

3.7 Conclusion

In this section, we discuss the ways in which our logic improves on previous frameworks dealing with the relationship between knowledge and action; we also discuss the limitations of the system.

3.7.1 Time and Action

The temporal part of our logic is simple and very expressive. One can make both eternal and indexical temporal assertions, as well as express relations between indexically specified and objectively specified times. It also meshes well with the epistemic part of the logic. The fact that time is reified (i.e., that there are temporal terms) allows quantification over times into epistemic contexts, so one can make very weak ascriptions of temporal knowledge; for example, one can say that an agent knows that another agent knows what time it is without himself knowing it. One can assert that actions are done over arbitrary time intervals, so it is possible to state that several actions are being done concurrently or that some continuous process is happening. It is also possible to talk about actions involving several agents. We do not exploit all this expressiveness, as there is already sufficient difficulty in accounting for (indexical) knowledge prerequisites of simple sequential actions involving a single agent; but its presence makes the framework easier to extend or adapt to other applications. There is one major limitation: in contrast to dynamic logic, we do not provide a way of expressing the occurrence of actions involving indefinite iteration (unbounded while-loops).

3.7.2 Knowledge

What are the advantages of our account of knowledge over previous accounts? Firstly, it does capture the distinction between indexical and *de re* knowledge: an agent may know something in an indexical way without knowing it *de re* and vice-versa. Moreover our account fully handles time, in particular, temporally indexical knowledge. The account has a simple syntax which makes it easy to make assertions involving indexical knowledge. Its model-theoretic semantics is a simple and natural extension of standard possible-world semantics. It retains what is perhaps the most important feature of possible-world semantics: the correspondence between constraints on the accessibility relation and important properties of the notion formalized. Finally, we have proposed a formalization of the requirement that knowledge be persistent that works well with both temporally indexical and temporally absolute knowledge.

There are several limitations to our account of knowledge. Since it is based on standard possible-world semantics, it only accounts for a very idealized notion of knowledge, a notion that ascribes logical omniscience to agents (i.e., knowledge of all validities and knowledge of all logical consequences of one's knowledge). It is also the case that our interpretation of *de re* knowledge as requiring knowledge of an objective standard name for the object does not always fit well with intuitions about the notion. But this is a problem that we have in common with all logics of knowledge, as there seems to be no commonly agreed upon theory of *de re* knowledge. More work is clearly required on this front, but since the relation between knowledge and action is crucial to our understanding of the notion, it can be hoped that our account and applications of it to various domains can lead to new insights into the question (see Grove and Halpern (1989) for an interesting attempt at formal analysis).

Finally, it would also be desirable to have a more thorough axiomatization of the logic, a complete one if that is possible. This would provide a more explicit characterization of the system, an immediately understandable account of the logic's properties. It might also be a step towards doing theorem-proving in the logic, which would have many applications. The set of properties identified in section 3.5 would provide a good starting point. Developing an axiomatization should not be an insurmountable task since most aspects of the logic are based on previously axiomatized systems; but handling interactions between these elements may be a difficult task.

Before moving on, we should also mention some alternatives to our formalization of indexical knowledge. In our logic, the interpretation of formulas is context dependent. Instead, one could have required this context dependence to be made explicit: the agent and time of the context could have been represented by ordinary variables, which are either bound directly by the **Know** operator or lambda-abstracted. For example, to say that agent θ^a knows at time θ^t that he himself is currently thirsty, we write $\mathbf{At}(\theta^t, \mathbf{Know}(\theta^a, \textsc{Thirsty}))$ or $\mathbf{At}(\theta^t, \mathbf{Know}(\theta^a, \textsc{Thirsty}(\mathbf{self})))$; instead of that one might want to write something like $\mathbf{Know}_{a,t}(\theta^a, \theta^t, \textsc{Thirsty}(a,t))$, or $\mathbf{Know}(\theta^a, \theta^t, \lambda(a,t).\textsc{Thirsty}(a,t))$, where a and t represent the agent and time of the context (Richard (1983) follows the latter approach). This could be done without changing our semantics for knowledge in any essential way. It would perhaps clarify the fact that our account is compatible with Lewis's view that having knowledge involves self-ascription of properties (Lewis 1979). But, it would have made for much heavier syntax and, given our modal treatment of knowledge, the approach followed is quite natural. Note that these alternative approaches could also be applied to the formalization of action and ability.

3.7.3 Ability

Our formalization of ability is an improvement over previous accounts (and in particular, Moore's) in several ways. Firstly, it does not require an agent to know who he is in order to be able to achieve a goal by doing an action. This is accomplished by having all references to the agent within the knowledge required be indexical references. Mere *de re* knowledge is neither necessary nor sufficient for the agent to be able to achieve the goal. We give a concrete example of this in the next chapter. As argued earlier, situations where an agent does not know who he is (in the sense of not knowing an objective standard name for himself) may not be common, but they should nonetheless be handled, since they can actually arise, they are at the root of the phenomenon of indexicality, and the framework required to handle other aspects of indexicality will also work for them.

A second way in which our account improves over previous work arises from the fact that it is based upon a logic that is more expressive with respect to temporal knowledge. The temporal and epistemic elements of our system do distinguish between temporally indexical knowledge and *de re* knowledge of times; they accommodate both kinds of knowledge; and they allow the two kinds of knowledge to be interrelated, for example, one can talk about an agent knowing what time it is. In section 5.3, we show that it is possible within the current framework to handle cases of actions with temporal knowledge prerequisites or effects (e.g., looking at one's watch) and actions that refer to absolute times (e.g., locking up at 5 p.m.).

Finally, other benefits accrue from the basing of our formalization upon a logic that includes a proper treatment of indexical knowledge in general. It sets the stage for the approach to the formalization of actions developed in the next chapter, whose main point is the minimization of the knowledge requirements of actions through the judicious use of indexical knowledge.

There are many limitations to our formalization of ability. Firstly, it is limited by the basic assumptions of the models we are working under and a high degree of idealization in the notions involved. The model of action on which our formalization is based is essentially the "planning" paradigm, which comes with all sort of strong assumptions, such as, the notion that action is thoroughly under rational control. The account of knowledge that we are using involves idealizations that yield properties such as consequential closure and absolute persistence, properties that simplify the formalization, but that also widen the gap that separate it from the phenomena it seeks to model. The formalization of ability itself, with its requirement that the action absolutely guarantee that the goal will be achieved, also brings its share of idealization. But how to relax any of these assumptions is a difficult research question in its own right and it seems impossible to avoid making this kind of simplification if we are to obtain any kind of formal account of the relation between indexical knowledge and action. As pointed out earlier, the treatment of indexical knowledge in relation to action in itself constitutes a major relaxation of the limitations associated with the standard view of knowledge and action.

The formalization only handles a basic class of actions, that of deterministic sequential single-agent actions without indefinite iteration (unbounded while-loops). This leaves plenty of room for extensions. The case of indefinite iteration should definitely be looked at. One could also extend coverage along standard directions, so as to handle classes of concurrent actions or plans involving multiple agents. Perhaps of a deeper significance would be extensions that exploit the additional expressive power of our formalism with respect to temporal knowledge: actions with temporal knowledge effects or prerequisites, or actions referring to absolute times; in general, actions that involve patterns of temporal reference (to both absolute and relative times) that go beyond the simple ones currently treated. As mentioned above, we take a stab at handling some of these in section 5.3 and we find that many can be handled within the current framework, albeit not as elegantly as one might want. More work on this is needed.

Finally, as mentioned previously, it would be nice to have a complete or at least thorough axiomatization of the logic.

Chapter 4

Robotics Applications

4.1 Introduction

In the previous chapter, we developed a theory of indexical knowledge, action, and ability that was claimed to form an adequate framework for the formalization of actions involving indexical knowledge prerequisites or effects. In this and the following chapter, we back up this claim by showing how various domains involving such actions can be formalized within the theory and how given an adequate formalization of a domain, one can prove that an agent is able to achieve certain goals by doing certain actions.

As will become clear, the framework we provide does not turn the formalization of actions which may have indexical knowledge prerequisites or effects into a trivial task. Many decisions must still be made in developing a formalization that have a crucial bearing on its adequacy from the point of view how much and what kind of knowledge it requires from an agent; for example, decisions as to which parameters a procedure should take. What we provide on this front is some general advice on what to watch for, as well as, the exemplary value of the domains that we formalize. Our central admonition in this respect is that *one should be careful not to impose excessive knowledge requirements upon agents in formalizing actions; in particular, one should merely require indexical knowledge, as opposed to de re knowledge, when that is sufficient.* Together with our formalization of the various domains, we provide a discussion of how this guideline has been put into practice. We also discuss various issues that arise in formalizing the knowledge prerequisites and effects of actions, in particular, what place *de re* knowledge should have in the result.

4.2 A Robotics Domain

We will now use the theory to formalize aspects of a simple robotics domain and show how the resulting formalization can be used to prove various statements concerning the ability of agents to achieve goals by doing actions. We will argue that our framework allows a much more accurate modeling of these situations than frameworks that ignore indexicality. Our domain involves a robot, call him ROB, that moves about on a two-dimensional grid. Since our purpose is not to model complex patterns of interactions, but to present and justify our account of indexical knowledge and action, our formalization will be based on the assumption that the robot is the only source of activity in the domain. We take our robot to have the following repertory of basic actions (primitives of his architecture): he may move forward by one square, he may turn right or left 90°, he may sense whether an object is on the square where he is currently positioned and if there is one, what shape it has, and he may pick up an object from the current square or put down the object he is holding on the current square. It should be clear that in spite of the simplicity of this domain, it contains analogues to a large number of problems encountered in planning actual robot navigation, manipulation, and perception. For instance, one can view objects of particular shapes as landmarks and the robot can then navigate by recognizing such landmarks. We assume that there are no physical obstacles to the robot's movements; in particular, an object being on a square does not prevent the robot from being on it too (one can imagine the robot as standing over the object).

4.3 Ability to Manipulate an Object

The indexicality of action manifests itself in many ways in this domain. One key way is that a robot can act upon (manipulate) an object as long as he knows where that object is *relative to himself*; he need not know either the object's absolute position or his own. First consider a simple instance of this where the robot wants to pick up an object and is actually positioned where that object is. Relevant aspects of the domain are formalized by making various assumptions,[1] most of which have to do with the types of action involved.

[1] Assumptions are essentially like non-logical axioms in a theory. Thus, we will only deal with semantic structures where the assumptions come out true. Note that due to this, assumptions not only hold at time now, but at all times, and it is common knowledge that this is the case.

The following assumption specifies the effects of the action PICKUP:

Assumption 4.1 (Effects of PICKUP)

$$\models \forall x(\text{OBJECT}(x) \wedge \text{POS}(x) = \textbf{here} \wedge \neg\exists y\ \text{HOLDING}(y) \supset \textbf{Res}(\text{PICKUP}, \text{HOLDING}(x)))$$

It says that if some object x is positioned where the agent currently is and he is not currently holding anything, then his doing the action PICKUP next will result in his holding x.[2] This means that under these conditions, it is both physically possible for him to do PICKUP, and his doing so necessarily results in his holding the object. In fact we assume that all basic actions are always possible. The view adopted is that such actions characterize essentially internal events which may have various external effects depending on the circumstances.[3] We also assume that agents always know how to do basic actions, that is, know what primitive actions they denote. This is formalized as follows:

Assumption 4.2 (Basic actions are known)

$$\models \exists d\ \textbf{Know}(d = \theta), \text{ where } \theta \text{ is any basic action constant}$$

We omit the frame assumptions for PICKUP, which say that it does not affect the position or orientation of anything and that unheld objects that are not where the agent is remain unheld.

Now clearly, just having *de re* knowledge of some object (i.e., $\exists x\ \textbf{Know}(\text{OBJECT}(x))$) is insufficient for being able to pick it up; something must be known about the object's position. If we only wanted to model the agent at a high level of abstraction, we might be willing to assume that as soon as an agent knows which object is involved, he would know how to get to it (or how to find out). But there clearly are circumstances where such assumptions are invalid and modeling at such an abstract level would leave out a great deal about how action is actually produced. We want an account that addresses the issue of what information the agent must exploit in order to be able to get at the object.

[2]Note that even though we are specifically talking about the agent of the context in the above, the attribution in fact applies to all agents, since it is assumed that the assertion is valid, that is, satisfied at all indices (it is a property of our logic that if $\models \varphi$, then $\models \forall a \textbf{By}(a, \varphi)$). This is not a problem because we are assuming that there are no interactions between agents in this domain.

[3]Note that assumption 4.1 only specifies what happens when PICKUP is done under the conditions stated. What its effects are in other circumstances is not addressed.

In a discussion of the robot action of "putting a block on another block", Moore (1985) recognizes this and suggests that it be defined in terms of lower-level actions involving arm motions to the objects' positions, grasping, and ungrasping. Now, knowledge of an object's absolute position is not sufficient for being able to act upon it (and nor is it necessary). One may not know what one's absolute position and orientation is and therefore may not be able to deduce where the object is relative to oneself. Our formalization reflects this fact. For instance, one can prove the following proposition with respect to the simple situation discussed earlier. It says that even if the agent is currently at some position $\vec{p}$ and knows that the absolute position of some object is $\vec{p}$ and that he is not holding anything, he still might not be able to achieve the goal of holding some object by doing the action PICKUP. The reason for this is simply that he may not know that he is at $\vec{p}$.

Proposition 4.1

$$\not\models \exists \vec{p}(\mathbf{here} = \vec{p} \wedge \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \vec{p}) \wedge \neg\exists y\, \textsc{Holding}(y))) \\ \supset \mathbf{Can}(\textsc{pickup}, \exists x\, \textsc{Holding}(x))$$

Proof:
We produce a structure that falsifies the formula.
1. Let an index e in some semantic structure and an assignment g be such that

$$\mathrm{e,g} \models \mathbf{here} = \vec{p} \wedge \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \vec{p}) \wedge \neg\exists x\, \textsc{Holding}(x))$$

Let an index e′ be such that $\langle \mathrm{e}, \mathrm{e}' \rangle \in \mathsf{K}$, $[\![\mathbf{here}]\!]_{\mathrm{e}',\mathrm{g}} \neq \mathrm{g}(\vec{p})$, and

$$\mathrm{e}', \mathrm{g} \models \mathbf{Does}(\textsc{pickup}, t) \wedge \neg\mathbf{At}(t, \exists x\, \textsc{Holding}(x))$$

Note that this is consistent with assumptions 4.1 and 4.2. It is easy to verify that this satisfies all our semantic constraints.
2. Now suppose that $\mathrm{e,g} \models \mathbf{Can}(\textsc{pickup}, \exists x\, \textsc{Holding}(x))$. By the definition of **Can**, this means that

$$\mathrm{e,g} \models \exists d\, \mathbf{Know}(d = \textsc{pickup} \wedge \mathbf{Res}(\textsc{pickup}, \exists x\, \textsc{Holding}(x)))$$

which implies that for all e′ such that $\langle \mathrm{e}, \mathrm{e}' \rangle \in \mathsf{K}$, it must be the case that

$$\mathrm{e}', \mathrm{g} \models \mathbf{Res}(\textsc{pickup}, \exists x\, \textsc{Holding}(x)))$$

By the definition of **Res** and **AfterNec**, this implies that

$$\mathrm{e}', \mathrm{g} \models \Box\forall t(\mathbf{Does}(\textsc{pickup}, t) \supset \mathbf{At}(t, \exists x\, \textsc{Holding}(x)))$$

Since $\approx_t$ is reflexive, we must then have,

$$e', g \models \mathbf{Does}(\textsc{pickup}, t) \supset \mathbf{At}(t, \exists x\, \textsc{Holding}(x))$$

which contradicts the stipulation in step 1. ■

On the other hand, we can also prove that if the agent knows that some object is *where he currently is* and that he is not holding anything, then he must be able to achieve the goal of holding some object by doing PICKUP:

Proposition 4.2

$$\begin{aligned}&\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}) \wedge \neg\exists y\, \textsc{Holding}(y))\\ &\quad \supset \mathbf{Can}(\textsc{pickup}, \exists x\, \textsc{Holding}(x))\end{aligned}$$

Proof:
1. Let's assume that

$$e, g \models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}) \wedge \neg\exists y\, \textsc{Holding}(y))$$

that is, that for all $e' = \langle w', a', t' \rangle$, $\langle e, e' \rangle \in K$,

$$e', g \models \exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}) \wedge \neg\exists y\, \textsc{Holding}(y)$$

We must show that $e, g \models \mathbf{Can}(\textsc{pickup}, \exists x\, \textsc{Holding}(x))$. By the definition of **Can**, this means that

$$e, g \models \exists d\, \mathbf{Know}(d = \textsc{pickup} \wedge \mathbf{Res}(\textsc{pickup}, \exists x\, \textsc{Holding}(x)))$$

that is, that there exists a d, such that for all $e', \langle e, e' \rangle \in K$,

$$e', g_d \models d = \textsc{pickup} \wedge \mathbf{Res}(\textsc{pickup}, \exists x\, \textsc{Holding}(x))$$

where $g_d = g\{d \rightarrow \mathrm{d}\}$.
2. Let's first show that there exists a d, such that for all $e', \langle e, e' \rangle \in K$, $e', g_d \models d = \textsc{pickup}$ (i.e., that the agent knows what the action is). By assumption 4.2, we must have that $e, g \models \exists d\, \mathbf{Know}(d = \textsc{pickup})$, and so the above must hold.
3. So what remains to be proven is that $e', g_d \models \mathbf{Res}(\textsc{pickup}, \exists x\, \textsc{Holding}(x))$ (i.e., that the agent knows that his doing PICKUP next results in the goal being achieved afterwards). We must use assumption 4.1, which says that

$$\models \forall x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here} \wedge \neg\exists y\, \textsc{Holding}(y) \supset \mathbf{Res}(\textsc{pickup}, \textsc{Holding}(x)))$$

By step 1, we have that for all e', $\langle e, e' \rangle \in K$, there exists an x such that

$$e', g_x \models \textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here} \wedge \neg\exists y\, \textsc{Holding}(y)$$

where $g_x = g\{x \to x\}$. So by assumption 4.1, we have that

$$e', g_x \models \mathbf{Res}(\textsc{pickup}, \textsc{Holding}(x))$$

By proposition 3.69, this implies that $e', g_d \models \mathbf{Res}(\textsc{pickup}, \exists x\, \textsc{Holding}(x))$. ■

The agent can be totally ignorant of what his (and the object's) absolute position is and still be able to achieve the goal.

Note that proposition 4.2 makes no requirement that the object that the agent ends up holding be the same as the one that was at his position before the action. This may appear too weak and an easy fix would involve assuming that the agent knows which object is involved. But it is possible to strengthen the above proposition without requiring such *de re* knowledge. For example, the following proposition captures the fact that the agent knows that after the action, he would be holding some object that was where he was before doing the action. Specifically, it says that if the agent knows that some object is currently at his position and that he is not currently holding anything, then he can by doing action PICKUP achieve the goal of holding some object that was at his own position before the PICKUP he has just done.

Proposition 4.3

$$\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}) \wedge \neg\exists y\, \textsc{Holding}(y)) \supset$$
$$\mathbf{Can}(\textsc{pickup}, \exists x(\textsc{Holding}(x) \wedge \mathbf{DoneWhen}(\textsc{pickup}, \textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here})))$$

Proof:
1. Let $\textsc{Oh}(x) \overset{\text{def}}{=} \textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}$. Assume that $e, g \models \mathbf{Know}(\exists x(\textsc{Oh}(x)) \wedge \neg\exists y\, \textsc{Holding}(y))$, that is, that for all $e' = \langle w', a', t' \rangle$, such that $\langle e, e' \rangle \in K$, $e', g \models \exists x(\textsc{Oh}(x)) \wedge \neg\exists y\, \textsc{Holding}(y)$. Let $\varphi_g \overset{\text{def}}{=} \exists x(\textsc{Holding}(x) \wedge \mathbf{DoneWhen}(\textsc{pickup}, \textsc{Oh}(x)))$. We must show that $e, g \models \mathbf{Can}(\textsc{pickup}, \varphi_g)$. By the definition of **Can**, this requires showing that

$$e, g \models \exists d\, \mathbf{Know}(d = \textsc{pickup} \wedge \mathbf{Res}(\textsc{pickup}, \varphi_g))$$

that is, that there exists a d, such that for all $e', \langle e, e' \rangle \in K$,

$$e', g_d \models d = \textsc{pickup} \wedge \mathbf{Res}(\textsc{pickup}, \varphi_g)$$

where $g_d = g\{d \to d\}$.
2. By assumption 4.2, we must have that $e, g \models \exists d\, \mathbf{Know}(d = \textsc{pickup})$, and so it must be the case that there exists a d, such that for all $e', \langle e, e' \rangle \in K$, $e', g_d \models d = \textsc{pickup}$.
3. So what remains to be proven is that $e', g_d \models \mathbf{Res}(\textsc{pickup}, \varphi_g)$. By 1, we have that for all e', such that $\langle e, e' \rangle \in K$, there exists an x such that

$$e', g_x \models \textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here} \wedge \neg\exists y\, \textsc{Holding}(y)$$

where $\mathrm{g_x} = \mathrm{g}\{x \to \mathrm{x}\}$. So by assumption 4.1, we have that

$$\mathrm{e}', \mathrm{g_x} \models \mathbf{Res}(\textsc{pickup}, \textsc{Holding}(x))$$

Note that this implies that $\mathbf{PhyPoss}(\textsc{pickup})$.
4. By 3 and proposition 3.58, we must have that

$$\mathrm{e}', \mathrm{g_x} \models \Box(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here})$$

So by proposition 3.71, it must be the case that

$$\mathrm{e}', \mathrm{g_x} \models \mathbf{AfterNec}(\textsc{pickup}, \mathbf{DoneWhen}(\textsc{pickup}, \textsc{Oh}(x)))$$

5. By proposition 3.66, this combines with 3 to yield that

$$\mathrm{e}', \mathrm{g_x} \models \mathbf{Res}(\textsc{pickup}, \textsc{Holding}(x) \wedge \mathbf{DoneWhen}(\textsc{pickup}, \textsc{Oh}(x)))$$

By proposition 3.69, this implies that $\mathrm{e}', \mathrm{g_d} \models \mathbf{Res}(\textsc{pickup}, \varphi_g)$. ■
This can be strengthened further to require uniqueness. But it should be clear that identifying the objects involved in the initial and goal situations, without requiring that it be known what objects they are, is not a trivial matter.

Before moving on, let's examine another variant of this situation. First, imagine that an agent a knows that there is an object where he himself is and that he is not holding anything. Then a is able to achieve the goal of holding something by doing PICKUP. Formally, if we let φ be the formula of proposition 4.2, then $\models \mathbf{By}(a, \varphi)$. However, if we imagine that a instead knows that there is an object where a is, it no longer follows that he is able to achieve the goal.

Proposition 4.4

$$\not\models \mathbf{By}(a, \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \textsc{pos}(a)) \wedge \neg\exists y\ \textsc{Holding}(y))$$
$$\supset \mathbf{Can}(\textsc{pickup}, \exists x\ \textsc{Holding}(x))$$

The reason why this is not valid is simply that a may not know that he is a. This shows that knowing of oneself (*de re*) that if one does the action, the goal will necessarily hold afterwards, as Moore's formalization of ability requires, is not sufficient for ability. One can similarly show that such *de re* knowledge is not necessary either (in some models of proposition 4.2, the agent does not have such knowledge).

4.4 Going to a Relative Position

Let us now look at navigation. Since there are no obstacles in our robot's world, it seems that given any relative position, the robot should be able to go there. Of course, he may not be able to go to a given absolute position, if he does not know where he is. Let us show formally that this is the case. We first consider the restricted case where the robot wants to go to a position directly in front of him.

The effects of action FORWARD are specified as follows:

Assumption 4.3 (Effects of FORWARD)

$$\models \forall \vec{p} \forall o(\mathbf{here} = \vec{p} \wedge \mathbf{selfori} = o \supset \mathbf{Res}(\textsc{forward}, \mathbf{here} = \vec{p} + \langle 1, 0 \rangle \times \textsc{rot}(o)))$$

Definition 4.1 $\mathbf{selfori} \overset{\text{def}}{=} \textsc{ori}(\mathbf{self})$

Definition 4.2 $\textsc{rot}(\theta_o^i) \overset{\text{def}}{=} \left\langle \begin{array}{cc} \cos\theta_o^i & \sin\theta_o^i \\ -\sin\theta_o^i & \cos\theta_o^i \end{array} \right\rangle$

Assumption 4.4 $\models \forall x(\textsc{Holding}(x) \supset \textsc{pos}(x) = \mathbf{here})$

Assumption 4.3 says that as a result of doing FORWARD, the agent moves one square further along the direction he is facing; **selfori** represents the orientation of the agent with respect to the absolute frame of reference and $\textsc{rot}(\theta_o^i)$ is the rotation matrix associated with angle θ_o^i. Assumption 4.4 says that objects held by the agent are where he is.

We also need the following frame assumptions:

Assumption 4.5 (FORWARD does not affect selfori)

$$\models \forall o(\mathbf{selfori} = o \supset \mathbf{AfterNec}(\textsc{forward}, \mathbf{selfori} = o))$$

Assumption 4.6 (FORWARD does not affect POS of unheld objects)

$$\models \forall \vec{p} \forall x(\textsc{pos}(x) = \vec{p} \wedge \neg\textsc{Holding}(x) \supset \mathbf{AfterNec}(\textsc{forward}, \textsc{pos}(x) = \vec{p}))$$

Assumption 4.7 (FORWARD does not affect ¬HOLDING)

$$\models \forall x(\neg\textsc{Holding}(x) \supset \mathbf{AfterNec}(\textsc{forward}, \neg\textsc{Holding}(x)))$$

Assumption 4.5 states that after doing FORWARD, the agent's orientation must remain unchanged. Assumption 4.6 says that after the agent does FORWARD, the position of objects that are not held by him must remain the same as before. Assumption 4.7 states that objects that are not held by the agent must remain unheld after he does FORWARD.

We will also need the following assumption; it says that any overlapping instances of the FORWARD action must be the same instance (in the sense that they have the same endpoints).

Definition 4.3 $\text{OVERLAPS}(\theta_{s1}, \theta_{e1}, \theta_{s2}, \theta_{e2}) \overset{\text{def}}{=} \theta_{s1} < \theta_{e2} \wedge \theta_{s2} < \theta_{e1}$

Assumption 4.8 (FORWARD is solid)

$$\models \forall t_{s1} \forall t_{e1} \forall t_{s2} \forall t_{e2} (\mathbf{DoneFromTo}(\text{FORWARD}, t_{s1}, t_{e1}) \wedge \mathbf{DoneFromTo}(\text{FORWARD}, t_{s2}, t_{e2})$$
$$\wedge\ \text{OVERLAPS}(t_{s1}, t_{e1}, t_{s2}, t_{e2}) \supset t_{s1} = t_{s2} \wedge t_{e1} = t_{e2})$$

Actions that have this property are called "solid" by Shoham (1987).

One will have noticed that we use expressions from the language of arithmetics and analytic geometry in formalizing this domain. We make the following assumptions regarding these forms:

Assumption 4.9
All the constants and function symbols from arithmetics and analytic geometry used are rigid.

Assumption 4.10

$\models \forall v \varphi \supset \varphi\{v \mapsto \theta\}$, provided that θ is free for v in φ and all constants and function symbols θ contains are arithmetic or from analytic geometry

Assumption 4.11
All facts from arithmetics and analytic geometry are valid.

Assumption 4.10 essentially say that the usual restrictions on specialization do not apply to arithmetic or analytic geometry expressions. It should be easily provable from assumption 4.9.

Finally, let us add a few more definitions. RPOSTOAPOS is a function that converts a relative position into an absolute one; similarly, RORITOAORI converts a relative orientation to an absolute one. $\textsc{MovedToBy}(\theta_p^i, \theta_o^i, \delta)$ means that the agent has just moved to relative position θ_p^i and relative orientation θ_o^i by doing action δ.

Definition 4.4 $\textsc{rposToApos}(\theta_p^i) \stackrel{\text{def}}{=} (\mathbf{here} + \theta_p^i \times \textsc{rot}(\mathbf{selfori}))$

Definition 4.5 $\textsc{roriToAori}(\theta_o^i) \stackrel{\text{def}}{=} \textsc{mod}_{2\pi}(\mathbf{selfori} + \theta_o^i)$

Definition 4.6

$$\begin{aligned}&\textsc{MovedToBy}(\theta_p^i, \theta_o^i, \delta) \stackrel{\text{def}}{=}\\ &\quad \forall v_p^i \forall v_o^i (\mathbf{DoneWhen}(\delta, v_p^i = \textsc{rposToApos}(\theta_p^i) \wedge v_o^i = \textsc{roriToAori}(\theta_o^i)) \supset\\ &\qquad\quad \mathbf{here} = v_p^i \wedge \mathbf{selfori} = v_o^i)),\end{aligned}$$

where v_p^i and v_o^i do not occur in θ_p^i, θ_o^i, or δ

Let us now proceed with the proof that the robot can go to any relative position that is directly in front of him. First, we prove the following proposition, which says that doing $\textsc{forward}$ results in the agent's being on the square that was at relative position $\langle 1,0\rangle$ before the action and that his orientation is unchanged.

Proposition 4.5 $\models \mathbf{Res}(\textsc{forward}, \textsc{MovedToBy}(\langle 1,0\rangle, 0, \textsc{forward}))$

Proof:
1. Let us first show that $\models \mathbf{PhyPoss}(\textsc{forward})$. By assumption 4.3 and the definition of **Res**, we have that

$$\models \exists\vec{p}\exists o(\mathbf{here} = \vec{p} \wedge \mathbf{selfori} = o) \supset \mathbf{PhyPoss}(\textsc{forward})$$

By proposition 3.6, the desired result must follow.
2. Let us now show that $\models \mathbf{AfterNec}(\textsc{forward}, \textsc{MovedToBy}(\langle 1,0\rangle, 0, \textsc{forward}))$. Take arbitrary $\mathsf{e} = \langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle$ and w^* and t_e such that $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$. Let $\mathsf{e}^* = \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}\rangle$ and $\mathsf{e}_e^* = \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_e\rangle$. Assume that $\Delta([\![\textsc{forward}]\!]_{\mathsf{e}^*,\mathsf{g}}, \mathsf{e}^*, \mathsf{t}_e)$.
3. Assume that

$$\mathsf{e}_e^*, \mathsf{g} \models \mathbf{DoneWhen}(\textsc{forward}, \vec{p} = \textsc{rposToApos}(\langle 1,0\rangle) \wedge o = \textsc{roriToAori}(0))$$

We must now show that $\mathsf{e}_e^*, \mathsf{g} \models \mathbf{here} = \vec{p} \wedge \mathbf{selfori} = o$.
4. 3 means that

$$\begin{aligned}\mathsf{e}_e^*, \mathsf{g} \models\ & \exists t(\mathbf{DoneFrom}(\textsc{forward}, t) \wedge\\ &\quad \mathbf{At}(t, \vec{p} = \textsc{rposToApos}(\langle 1,0\rangle) \wedge o = \textsc{roriToAori}(0)))\end{aligned}$$

By 2 and assumption 4.8, we must then have that

$$\mathsf{e}^*, \mathsf{g} \models \vec{p} = \textsc{rposToApos}(\langle 1,0\rangle) \wedge o = \textsc{roriToAori}(0)$$

5. By assumption 4.5 and the reflexivity of $\approx_t$ (constraint 3.4), we must have that $[\![\textbf{selfori}]\!]_{e^*_e,g} = [\![\textbf{selfori}]\!]_{e^*,g}$. And by the definition of RORITOAORI, this implies that $[\![\textbf{selfori}]\!]_{e^*_e,g} = [\![\text{RORITOAORI}(0)]\!]_{e^*,g}$ (we assume that $\models 0 \leq \text{ORI}(\theta^a) < 2\pi$).
6. By assumption 4.3 and the reflexivity of $\approx_t$, we have that

$$[\![\textbf{here}]\!]_{e^*_e,g} = [\![\textbf{here}]\!]_{e^*,g} + \langle 1,0\rangle \times \text{ROT}([\![\textbf{selfori}]\!]_{e^*,g})$$

By the definition of RPOSTOAPOS, this means that $[\![\textbf{here}]\!]_{e^*_e,g} = [\![\text{RPOSTOAPOS}(\langle 1,0\rangle)]\!]_{e^*,g}$.
7. By 3, 4, 5, and 6, we must then have that

$$\begin{aligned} e^*_e, g \models &\textbf{DoneWhen}(\text{FORWARD}, \vec{p} = \text{RPOSTOAPOS}(\langle 1,0\rangle) \wedge o = \text{RORITOAORI}(0)) \\ &\supset \textbf{here} = \vec{p} \wedge \textbf{selfori} = o \end{aligned}$$

Since g is arbitrary, we must also have that

$$\begin{aligned} e^*_e, g \models \forall\vec{p}\forall o(&\textbf{DoneWhen}(\text{FORWARD}, \vec{p} = \text{RPOSTOAPOS}(\langle 1,0\rangle) \wedge o = \text{RORITOAORI}(0)) \\ &\supset \textbf{here} = \vec{p} \wedge \textbf{selfori} = o) \end{aligned}$$

that is, that $e^*_e, g \models \text{MOVEDTOBY}(\langle 1,0\rangle, 0, \text{FORWARD})$. Since e, w^*, t_e, and g are arbitrary, this together with 2 implies that

$$\models \textbf{AfterNec}(\text{FORWARD}, \text{MOVEDTOBY}(\langle 1,0\rangle, 0, \text{FORWARD}))$$

■

From this, we can now prove the desired result, which says that by doing FORWARD, one is able to achieve the goal of knowing that one is on the square that was at relative position $\langle 1,0\rangle$ before doing the action and that one's orientation is unchanged.

Proposition 4.6 $\models \textbf{Can}(\text{FORWARD}, \textbf{Know}(\text{MOVEDTOBY}(\langle 1,0\rangle, 0, \text{FORWARD})))$

Proof:
1. Let $\varphi \overset{\text{def}}{=} \text{MOVEDTOBY}(\langle 1,0\rangle, 0, \text{FORWARD})$. Take arbitrary e and g. We must show that $e, g \models \textbf{Can}(\text{FORWARD}, \textbf{Know}(\varphi))$. By the definition of **Can**, this amounts to showing that

$$e, g \models \exists d\, \textbf{Know}(d = \text{FORWARD} \wedge \textbf{Res}(\text{FORWARD}, \textbf{Know}(\varphi)))$$

that is, that there exists a d, such that for all e′, $\langle e, e'\rangle \in K$,

$$e', g_d \models d = \text{FORWARD} \wedge \textbf{Res}(\text{FORWARD}, \textbf{Know}(\varphi))$$

where $g_d = g\{d \to d\}$.
2. Let's first show that there exists a d, such that for all e′, $\langle e, e'\rangle \in K$, $e', g_d \models d = \text{FORWARD}$ (i.e., that the agent knows what the action is). By assumption 4.2, we must have that $e, g \models \exists d\, \textbf{Know}(d = \text{FORWARD})$, and so the above must hold.
3. So what remains to be proven is that $e', g_d \models \textbf{Res}(\text{FORWARD}, \textbf{Know}(\varphi))$ (i.e., that the agent knows that his doing FORWARD next results in the goal being achieved afterwards). By 1 and lemma 4.5, we must have that $e', g_d \models \textbf{Res}(\text{FORWARD}, \varphi)$. So by 2 and proposition 3.62, we have that $e', g_d \models \textbf{Res}(d, \varphi)$. Now since K is transitive, we must then have that $e', g_d \models \textbf{Know}(\textbf{Res}(d, \varphi))$. Then, by proposition 3.79, it follows

that $e', g_d \models \mathbf{Res}(d, \mathbf{Know}(\varphi))$. Finally, by 2 and proposition 3.62, we must have that $e', g_d \models \mathbf{Res}(\textsc{forward}, \mathbf{Know}(\varphi))$. ∎

It should be possible to extend this and prove the following conjecture;[4] it says that by doing FORWARD n times in sequence, one is able to achieve the goal of knowing that one is on the square that was at relative position $\langle n, 0\rangle$ before doing these actions and that one's orientation is unchanged.

Conjecture 4.1

$$\text{For all } n \in \mathbb{N},\ \models \mathbf{Can}(\textsc{forward}^n, \mathbf{Know}(\mathrm{MovedToBy}(\langle n,0\rangle, 0, \textsc{forward}^n)))$$

Now, let us look at the general case. The action of going to a relative position is defined below. The definition goes as follows: to go to relative position $\langle n, 0\rangle$, where $n \in \mathbb{N}$, one simply goes forward n times; to go to relative position $\langle n, 0\rangle$, where n is a negative integer, that is, to a position behind oneself, one turns left twice (i.e., 180°), then goes to relative position $\langle -n, 0\rangle$, and then one turns left twice again, so as to return to one's original orientation; finally, to go to an arbitrary relative position $\langle n, m\rangle$, one first goes to $\langle n, 0\rangle$, that is, the right position on the x axis, then one turns left 90°, then goes to $\langle m, 0\rangle$, and then turns left three times (i.e., 270°), to return to the original orientation.

Definition 4.7

$$\textsc{goRpos}(\langle n,0\rangle) \stackrel{\text{def}}{=} \textsc{forward}^n, \text{ where } n \in \mathbb{N}$$

$$\textsc{goRpos}(\langle -n,0\rangle) \stackrel{\text{def}}{=} \textsc{left}^2; \textsc{goRpos}(\langle n,0\rangle); \textsc{left}^2, \text{ where } n \in \mathbb{N}^+$$

$$\textsc{goRpos}(\langle n,m\rangle) \stackrel{\text{def}}{=} \textsc{goRpos}(\langle n,0\rangle); \textsc{left}; \textsc{goRpos}(\langle m,0\rangle); \textsc{left}^3, \text{ where } n, m \in \mathbb{Z}$$

It should now be fairly clear how a formalization for the actions of turning left and turning right would go, so we will omit it and simply make the following assumption, which says that by doing LEFT, one is able to achieve the goal of knowing that one's orientation is 90° to the left of what it was before doing the action and that one's position is unchanged.

Assumption 4.12 $\models \mathbf{Can}(\textsc{left}, \mathbf{Know}(\mathrm{MovedToBy}(\langle 0,0\rangle, \pi/2, \textsc{left})))$

[4]We label "conjecture" statements which we believe are provable (with perhaps minor adjustments), but for which we have no proof in hand. We discuss such conjectures in order to give a more complete view of the kinds of phenomena that can be formalized in our framework.

It should then be possible to show that by doing LEFT n times in sequence, one is able to achieve the goal of knowing that one's orientation is rotated by n times 90° to the left of what it was before doing the action and that one's position is unchanged.

Conjecture 4.2

$$\text{For all } \mathsf{n} \in \mathbb{N},\ \models \mathbf{Can}(\textsc{left}^{\mathsf{n}}, \mathbf{Know}(\textsc{MovedToBy}(\langle 0,0\rangle, \mathsf{n}\pi/2, \textsc{left}^{\mathsf{n}})))$$

Finally, from conjectures 4.2 and 4.1, it should follow that one can, by doing $\textsc{goRpos}(\langle \mathsf{n}, \mathsf{m}\rangle)$, achieve the goal of knowing that one is on the square that was at relative position $\langle \mathsf{n}, \mathsf{m}\rangle$ before doing these actions and that one's orientation is unchanged.

Conjecture 4.3

$$\begin{array}{l}\text{For all } \mathsf{n}, \mathsf{m} \in \mathbb{Z},\\ \models \mathbf{Can}(\textsc{goRpos}(\langle \mathsf{n}, \mathsf{m}\rangle), \mathbf{Know}(\textsc{MovedToBy}(\langle \mathsf{n}, \mathsf{m}\rangle, 0, \textsc{goRpos}(\langle \mathsf{n}, \mathsf{m}\rangle))))\end{array}$$

4.5 Picking Up an Object at a Relative Position

A common reason for going somewhere is that one wants to get hold of or manipulate something that is there. Thus, extending the results of the previous two sections, we note that knowing the relative position of an object is sufficient for being able to go and manipulate it. Let us now prove an instance of this kind of situation.

We must first make one additional assumption: that whether an entity is an object is not affected by any action.

Assumption 4.13 (actions don't affect $\textsc{Object}$)

$$\models \forall x(\textsc{Object}(x) \equiv \mathbf{AfterNec}(\theta^d, \textsc{Object}(x)))$$

We can now proceed with our proof. First, given the specifications of the action $\textsc{forward}$, we can show the following proposition, which says that if there is an object at position $\vec{p}$ relative to the agent and he is not holding anything, then his doing $\textsc{forward}$ must result in some object being at position $\vec{p} - \langle 1, 0\rangle$ relative to him and him still not holding anything. $\textsc{rpos}(x)$ represents the position of x relative to self; it is defined as follows:

Definition 4.8 $\textsc{rpos}(\theta^i) \overset{\text{def}}{=} ((\textsc{pos}(\theta^i) - \textbf{here}) \times \textsc{rot}(-\textbf{selfori}))$

Proposition 4.7

$$\models \forall\vec{p}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p}) \wedge \neg\exists y\, \textsc{Holding}(y) \supset$$
$$\textbf{Res}(\textsc{forward}, \exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p} - \langle 1, 0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y)))$$

Proof:
1. Assume that in some arbitrary structure

$$\mathrm{e}, \mathrm{g} \models \exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p}) \wedge \neg\exists y\, \textsc{Holding}(y)$$

that is, that there exists an x such that

$$\mathrm{e}, \mathrm{g_x} \models \textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p} \wedge \neg\exists y\, \textsc{Holding}(y)$$

where $\mathrm{g_x} = \mathrm{g}\{x \rightarrow \mathrm{x}\}$.
2. By assumption 4.3, we have that

$$\mathrm{e}, \mathrm{g} \models \textbf{here} = \vec{p_s} \wedge \textbf{selfori} = o \supset \textbf{Res}(\textsc{forward}, \textbf{here} = \vec{p_s} + \langle 1, 0\rangle \times \textsc{rot}(o))$$

From this, it follows that $\mathrm{e}, \mathrm{g} \models \textbf{PhyPoss}(\textsc{forward})$ and that

$$\mathrm{e}, \mathrm{g} \models \textbf{here} = \vec{p_s} \wedge \textbf{selfori} = o \supset \textbf{AfterNec}(\textsc{forward}, \textbf{here} = \vec{p_s} + \langle 1, 0\rangle \times \textsc{rot}(o))$$

3. Let $\mathrm{e} = \langle \mathrm{w}, \mathrm{a}, \mathrm{t}\rangle$. Take arbitrary w^* and $\mathrm{t_e}$ such that $\mathrm{w} \approx_{\mathrm{t}} \mathrm{w}^*$ and assume that $\Delta([\![\textsc{forward}]\!]_{\langle \mathrm{w}^*, \mathrm{a}, \mathrm{t}\rangle, \mathrm{g}}, \langle \mathrm{w}^*, \mathrm{a}, \mathrm{t}\rangle, \mathrm{t_e})$. Also let $\mathrm{e_e} = \langle \mathrm{w}^*, \mathrm{a}, \mathrm{t_e}\rangle$.
4. By 2, we have that $[\![\textbf{here}]\!]_{\mathrm{e_e},\mathrm{g}} = [\![\textbf{here}]\!]_{\mathrm{e},\mathrm{g}} + \langle 1, 0\rangle \times \textsc{rot}([\![\textbf{selfori}]\!]_{\mathrm{e},\mathrm{g}})$.
5. By assumption 4.5, we have that $[\![\textbf{selfori}]\!]_{\mathrm{e_e},\mathrm{g}} = [\![\textbf{selfori}]\!]_{\mathrm{e},\mathrm{g}}$.
6. By 1 and assumption 4.6, we have that $[\![\textsc{pos}(x)]\!]_{\mathrm{e_e},\mathrm{g_x}} = [\![\textsc{pos}(x)]\!]_{\mathrm{e},\mathrm{g_x}}$.

7. $[\![\textsc{rpos}(x)]\!]_{\mathrm{e_e},\mathrm{g_x}}$

$$\begin{aligned}
&= ([\![\textsc{pos}(x)]\!]_{\mathrm{e_e},\mathrm{g_x}} - [\![\textbf{here}]\!]_{\mathrm{e_e},\mathrm{g}}) \times \textsc{rot}(-[\![\textbf{selfori}]\!]_{\mathrm{e_e},\mathrm{g}}) \text{ by the definition of } \textsc{rpos} \\
&= ([\![\textsc{pos}(x)]\!]_{\mathrm{e},\mathrm{g_x}} - [\![\textbf{here}]\!]_{\mathrm{e_e},\mathrm{g}}) \times \textsc{rot}(-[\![\textbf{selfori}]\!]_{\mathrm{e},\mathrm{g}}) \text{ by 5 and 6} \\
&= ([\![\textsc{pos}(x)]\!]_{\mathrm{e},\mathrm{g_x}} - ([\![\textbf{here}]\!]_{\mathrm{e},\mathrm{g}} + \langle 1, 0\rangle \times \textsc{rot}([\![\textbf{selfori}]\!]_{\mathrm{e},\mathrm{g}}))) \times \textsc{rot}(-[\![\textbf{selfori}]\!]_{\mathrm{e},\mathrm{g}}) \text{ by 4} \\
&= ([\![\textsc{pos}(x)]\!]_{\mathrm{e},\mathrm{g_x}} - [\![\textbf{here}]\!]_{\mathrm{e},\mathrm{g}}) \times \textsc{rot}(-[\![\textbf{selfori}]\!]_{\mathrm{e},\mathrm{g}}) \\
&\quad -\langle 1, 0\rangle \times \textsc{rot}([\![\textbf{selfori}]\!]_{\mathrm{e},\mathrm{g}}) \times \textsc{rot}(-[\![\textbf{selfori}]\!]_{\mathrm{e},\mathrm{g}}) \\
&= [\![\textsc{rpos}(x)]\!]_{\mathrm{e},\mathrm{g_x}} - \langle 1, 0\rangle \text{ by definition of } \textsc{rpos}
\end{aligned}$$

8. By 1 and proposition 4.13, we have that $\mathrm{e_e}, \mathrm{g_x} \models \textsc{Object}(x)$.
9. By assumption 4.7, we have that if $\mathrm{e}, \mathrm{g} \models \neg\textsc{Holding}(y)$, then $\mathrm{e_e}, \mathrm{g} \models \neg\textsc{Holding}(y)$. Therefore by 1, we must have that $\mathrm{e_e}, \mathrm{g} \models \neg\exists y \textsc{Holding}(y)$ (note that $\models \forall y\varphi \supset \Box\forall y\varphi$).
10. So by 3, 7, 8, and 9, it must be the case that

$$\textbf{Res}(\textsc{forward}, \exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p} - \langle 1, 0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y))$$

■

Given the above result we can now show that if the robot knows that there is an object at position $\vec{p}$ relative to himself and that he is not holding anything, then by doing FORWARD he is able to achieve the goal that some object be at position $\vec{p} - \langle 1,0\rangle$ relative to himself and that he still not hold anything.

Proposition 4.8

$$\models \forall \vec{p}(\mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p}) \wedge \neg\exists y\, \textsc{Holding}(y)) \supset \\ \mathbf{Can}(\textsc{forward}, \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y))))$$

Proof:
1. Let's assume that in some arbitrary structure,

$$\mathbf{e},\mathbf{g} \models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p}) \wedge \neg\exists y\, \textsc{Holding}(y))$$

This means that for all $\mathbf{e}'$, such that $\langle \mathbf{e},\mathbf{e}'\rangle \in \mathsf{K}$,

$$\mathbf{e}',\mathbf{g} \models \exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p}) \wedge \neg\exists y\, \textsc{Holding}(y)$$

We must show that

$$\mathbf{e},\mathbf{g} \models \mathbf{Can}(\textsc{forward}, \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y)))$$

By definition 3.8, this amounts to showing that

$$\mathbf{e},\mathbf{g} \models \exists d\, \mathbf{Know}(d = \textsc{forward} \wedge \mathbf{Res}(\textsc{forward}, \\ \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y))))$$

that is, that there exists a $\mathbf{d}$, such that for all $\mathbf{e}', \langle \mathbf{e},\mathbf{e}'\rangle \in \mathsf{K}$,

$$\mathbf{e}',\mathbf{g}_{\mathbf{d}} \models d = \textsc{forward} \wedge \\ \mathbf{Res}(\textsc{forward}, \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y)))$$

where $\mathbf{g}_{\mathbf{d}} = \mathbf{g}\{d \rightarrow \mathbf{d}\}$.
2. Let's first show that there exists a $\mathbf{d}$, such that for all $\mathbf{e}', \langle \mathbf{e},\mathbf{e}'\rangle \in \mathsf{K}, \mathbf{e}',\mathbf{g}_{\mathbf{d}} \models d = \textsc{forward}$ (i.e., that the agent knows what the action is). By assumption 4.2, we must have that $\mathbf{e},\mathbf{g} \models \exists d\, \mathbf{Know}(d = \textsc{forward})$, and so the above must hold.
3. So what remains to be proven is that

$$\mathbf{e}',\mathbf{g}_{\mathbf{d}} \models \mathbf{Res}(\textsc{forward}, \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y)))$$

(i.e., that the agent knows that his doing FORWARD next results in the goal being achieved afterwards). By 1 and lemma 4.7, we must have that

$$\mathbf{e}',\mathbf{g}_{\mathbf{d}} \models \mathbf{Res}(\textsc{forward}, \exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y))$$

So by 2 and proposition 3.62, we have that

$$e', g_d \models \mathbf{Res}(d, \exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \land \neg\exists y\, \textsc{Holding}(y))$$

Now since K is transitive, we must then have that

$$e', g_d \models \mathbf{Know}(\mathbf{Res}(d, \exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \land \neg\exists y\, \textsc{Holding}(y)))$$

Then, by proposition 3.79, it follows that

$$e', g_d \models \mathbf{Res}(d, \mathbf{Know}(\exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \land \neg\exists y\, \textsc{Holding}(y)))$$

Finally, by 2 and proposition 3.62, we must have that

$$e', g_d \models \mathbf{Res}(\textsc{forward}, \mathbf{Know}(\exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{p} - \langle 1,0\rangle) \land \neg\exists y\, \textsc{Holding}(y)))$$

■

Then, we can prove by induction that if the robot knows that there is an object at position $\vec{p}$ relative to himself and that he is not holding anything, then by doing FORWARD n times in sequence, he is able to achieve the goal that some object be at position $\vec{p} - \langle n, 0\rangle$ relative to himself and that he still not hold anything.

Proposition 4.9

For all $n \in \mathbb{N}$,

$$\begin{aligned}&\models \forall\vec{p}(\mathbf{Know}(\exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{p}) \land \neg\exists y\, \textsc{Holding}(y)) \supset \\ &\qquad \mathbf{Can}(\textsc{forward}^n, \mathbf{Know}(\exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{p} - \langle n,0\rangle) \land \neg\exists y\, \textsc{Holding}(y))))\end{aligned}$$

Proof:
We prove this by induction over n.
Base case: $n = 0$. Then since $\delta^0 \overset{\text{def}}{=}$ **skip** and $\mathbf{Can}(\mathbf{skip}, \varphi) \overset{\text{def}}{=} \mathbf{Know}(\varphi)$, all we need to show is that

$$\begin{aligned}&\models \mathbf{Know}(\exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{p}) \land \neg\exists y\, \textsc{Holding}(y)) \supset \\ &\quad \mathbf{Know}(\mathbf{Know}(\exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{p} - \langle 0,0\rangle) \land \neg\exists y\, \textsc{Holding}(y)))\end{aligned}$$

This easily follows by proposition 3.14.
Induction step: Assume that the formula is valid for n and show that it must then be valid for $n + 1$. By the induction hypothesis and assumption 4.10, we have that

$$\begin{aligned}&\models \mathbf{Know}(\exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{q} - \langle 1,0\rangle) \land \neg\exists y\, \textsc{Holding}(y)) \supset \\ &\quad \mathbf{Can}(\textsc{forward}^n, \mathbf{Know}(\exists x(\textsc{Object}(x) \land \textsc{rpos}(x) = \vec{q} - \langle n+1,0\rangle) \land \neg\exists y\, \textsc{Holding}(y))))\end{aligned}$$

By proposition 3.80, it follows from this and proposition 4.8 that the formula is valid for $n + 1$. ■

Finally, we can chain this with our earlier result concerning ability to achieve holding an object by doing PICKUP (proposition 4.2), to obtain the following proposition; it says that if the robot knows that there is an object at position $\langle \mathsf{n}, 0\rangle$ relative to himself, that is, on the n'th square directly in front of him, and knows that he is not holding anything, then he is able to achieve the goal of holding some object by doing FORWARD n times followed by PICKUP.

Proposition 4.10

For all $\mathsf{n} \in \mathbb{N}$,

$$\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \langle \mathsf{n}, 0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y)) \supset \mathbf{Can}((\textsc{forward}^{\mathsf{n}}; \textsc{pickup}), \exists x\, \textsc{Holding}(x))$$

Proof:
By induction over n.
Base case: $\mathsf{n} = 0$.

$$\mathbf{Can}((\textsc{forward}^0; \textsc{pickup}), \varphi) \overset{\text{def}}{=} \mathbf{Can}((\mathbf{skip}; \textsc{pickup}), \varphi) \overset{\text{def}}{=} \mathbf{Know}(\mathbf{Can}(\textsc{pickup}, \varphi))$$

Thus, we must show that

$$\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \langle 0, 0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y)) \supset \mathbf{Know}(\mathbf{Can}(\textsc{pickup}, \exists x\, \textsc{Holding}(x)))$$

This easily follows from propositions 4.2 and 3.14.
Induction step: Similar that in the proof of proposition 4.9. ■

It should be straightforward to generalize this result and prove the following conjecture, which says that if the agent knows that there is an object at relative position $\langle \mathsf{n}, \mathsf{m}\rangle$ and that he is not holding anything, then he is able to achieve the goal of holding some object by first going to relative position $\langle \mathsf{n}, \mathsf{m}\rangle$ and then doing PICKUP.

Conjecture 4.4

For all $\mathsf{n}, \mathsf{m} \in \mathbb{Z}$,

$$\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \langle \mathsf{n}, \mathsf{m}\rangle) \wedge \neg\exists y\, \textsc{Holding}(y)) \supset \mathbf{Can}((\textsc{goRpos}(\langle \mathsf{n}, \mathsf{m}\rangle); \textsc{pickup}), \exists x\, \textsc{Holding}(x))$$

The proof would be similar to that of conjecture 4.3; it would require additional assumptions about action LEFT, namely that it does not affect the position of objects, as well as whether or not the agent is holding some object.

4.6 Perception

We will come back to this issue of what one *must* know in order to be able to go and manipulate an object, but now let's have a look at perception. As observed earlier, it too yields indexical knowledge.[5] In our domain, the action SENSE constitutes a limited form of perception. We formalize the effects of SENSE as follows:

Assumption 4.14

$$\models \mathbf{Res}(\text{SENSE}, \mathbf{Kwhether}(\exists x(\text{OBJECT}(x) \wedge \text{POS}(x) = \mathbf{here})))$$

Assumption 4.15

$$\begin{aligned} &\models \forall s(\neg\exists y \text{HOLDING}(y) \supset \\ &\qquad \mathbf{Res}(\text{SENSE}, \mathbf{Kwhether}(\exists x(\text{OBJECT}(x) \wedge \text{POS}(x) = \mathbf{here} \wedge \text{OFSHAPE}(x, s))))) \end{aligned}$$

Assumption 4.14 says that doing SENSE results in the agent knowing whether an object is present at his current position. Assumption 4.15 says that if the agent is not holding anything, then his doing SENSE results in him knowing whether there is an object of a given shape where he is.[6] From this and the fact that basic actions are assumed to be known, it follows trivially that by doing SENSE, the agent can find out if there is an object where he is and, if there is one, what its shape is.

The frame assumptions for sense are as follows:

Assumption 4.16 (SENSE does not affect POS)

$$\models \forall x \forall \vec{p}(\text{POS}(x) = \vec{p} \supset \mathbf{AfterNec}(\text{SENSE}, \text{POS}(x) = \vec{p})$$

[5]Davis (1988; 1989b; 1989a) has done interesting work on the topic of reasoning about knowledge and perception; however, he does not address the fact that the knowledge obtained from perception is indexical knowledge.

[6]One domain assumption that has been left unstated until now is that there is only room for a single object to be resting on a square. But we are also assuming that there are no obstacles to the movement of agents and that an agent can move onto a square even if there is an object on it (by standing above the object). Since the agent may be holding an object, this means that there are situations where two objects have the same position. This is why assumption 4.15 is restricted to situations where the agent is not holding anything; if he were holding something, the sensors might confuse the shapes of the held object with that of the object resting on the square.

Assumption 4.17 (SENSE does not affect selfori)

$$\models \forall o(\mathbf{selfori} = o \supset \mathbf{AfterNec}(\textsc{sense}, \mathbf{selfori} = o))$$

Assumption 4.18 (SENSE does not affect ¬HOLDING)

$$\models \forall x(\neg\textsc{Holding}(x) \supset \mathbf{AfterNec}(\textsc{sense}, \neg\textsc{Holding}(x)))$$

Finally, we assume that the shapes of objects are not affected by any action:

Assumption 4.19 (actions don't affect OFSHAPE)

$$\models \forall x \forall s(\textsc{OfShape}(x, s) \equiv \mathbf{AfterNec}(\theta^d, \textsc{OfShape}(x, s)))$$

4.7 Ability to Go Where an Object Is

Let's now go back to the issue of what one must know in order to be able to go and act upon an object. We said that knowing the relative position of the object was sufficient for this. But in real life, agents rarely know exactly what the relative locations of objects are. More typically, they know roughly where objects are and scan the general area until they find the object. For instance, if our robot knows that he is not holding anything and that there is an object either on the square where he is or on the one directly in front of him, then he can achieve the goal of being where an object is by first doing SENSE, and then doing FORWARD if the object turns out not to be where he is.[7]

Conjecture 4.5

For all $k \in \mathbb{N}$,

$$\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge (\textsc{pos}(x) = \mathbf{here} \vee \textsc{rpos}(x) = \langle 1, 0\rangle)) \wedge \neg\exists y \textsc{Holding}(y)) \supset$$
$$\mathbf{Can}((\textsc{sense}; \mathbf{ifThen}(\neg\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}), \textsc{forward})),$$
$$\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}))$$

[7]In this and the following example, the requirement that the agent know that he is not holding anything is not really necessary. But if the agent is holding something, then he should know that he is, and thus also know that the goal is already holding; so there would be no point in looking for an object. If on the other hand he is not holding anything, then he should know it.

The proposition below formalizes another instance; it says that if our robot knows that there is an object that is positioned at most k squares away along the row he is facing and that he is not holding anything, then he can get where some object is by sensing and then scanning forward, up to k squares, until he senses that an object is present. The $\text{SCAN}_{\mathsf{k}}(\varphi)$ action involves repetitively moving forward and sensing (up to k times) until φ becomes true.

Definition 4.9 $\text{SCAN}_{\mathsf{k}}(\varphi) \stackrel{\text{def}}{=} \textbf{while}_{\mathsf{k}}(\neg\varphi, \text{FORWARD}; \text{SENSE})$

Proposition 4.11

For all $\mathsf{k} \in \mathbb{N}$,

$$\models \textbf{Know}(\exists x(\text{OBJECT}(x) \wedge \exists n(\text{RPOS}(x) = \langle n, 0\rangle \wedge 0 \leq n \leq \mathsf{k})) \wedge \neg\exists y \text{HOLDING}(y)) \supset$$
$$\textbf{Can}((\text{SENSE}; \text{SCAN}_{\mathsf{k}}(\exists x(\text{OBJECT}(x) \wedge \text{POS}(x) = \textbf{here}))),$$
$$\exists x(\text{OBJECT}(x) \wedge \text{POS}(x) = \textbf{here}))$$

Let us now prove this proposition. First, let us introduce the following abbreviated forms:

Definition 4.10

$$\text{OH} \stackrel{\text{def}}{=} \exists x(\text{OBJECT}(x) \wedge \text{POS}(x) = \textbf{here})$$

$$\text{HS} \stackrel{\text{def}}{=} \exists y \text{HOLDING}(y)$$

$$\text{OIF}(\mathsf{k}) \stackrel{\text{def}}{=} \exists x(\text{OBJECT}(x) \wedge \exists n(\text{RPOS}(x) = \langle n, 0\rangle \wedge 0 \leq n \leq \mathsf{k})), \text{ where } \mathsf{k} \in \mathbb{N}$$

We first extend the result of proposition 4.7 into the following lemma, which says that if the robot is not holding anything, and there is an object positioned at most k + 1 squares away along the row he is facing, and that there is no object where he is, then his doing FORWARD must result in him not holding anything and in some object being positioned at most k squares in front of him.

Lemma 4.1

For all $\mathsf{k} \in \mathbb{N}$,

$$\models \text{OIF}(\mathsf{k}+1) \wedge \neg\text{OH} \wedge \neg\text{HS} \supset \textbf{Res}(\text{FORWARD}, \text{OIF}(\mathsf{k}) \wedge \neg\text{HS})$$

Proof:
1. Assume that in some arbitrary structure $\mathsf{e}, \mathsf{g} \models \mathsf{OIF}(\mathsf{k}+1) \wedge \neg\mathsf{OH} \wedge \neg\mathsf{HS}$. This implies that there exist x and n such that

$$\mathsf{e}, \mathsf{g_{xn}} \models \textsc{Object}(x) \wedge \textsc{rpos}(x) = \langle n, 0\rangle \wedge 1 \leq n \leq \mathsf{k}+1 \wedge \neg\mathsf{HS}$$

where $\mathsf{g_x} \overset{\text{def}}{=} \mathsf{g}\{x \to \mathsf{x}, n \to \mathsf{n}\}$.
2. Let $\mathsf{e} = \langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle$. Take arbitrary w^* and t_e and such that $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$ and assume that $\Delta([\![\textsc{forward}]\!]_{\langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}\rangle, \mathsf{g}}, \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}\rangle, \mathsf{t}_e)$. Also let $\mathsf{e}_e = \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_e\rangle$.
3. By 1 and proposition 4.7, we must have that

$$\mathsf{e}_e, \mathsf{g_{xn}} \models \textsc{Object}(x) \wedge \textsc{rpos}(x) = \langle n-1, 0\rangle \wedge \neg\mathsf{HS}$$

4. By 1, we must also have that $\mathsf{e}_e, \mathsf{g_{xn}} \models 1 \leq n \leq \mathsf{k}+1$. Combined with 3, this yields that $\mathsf{e}_e, \mathsf{g} \models \mathsf{OIF}(\mathsf{k}) \wedge \neg\mathsf{HS}$. ■

We can now prove the following lemma, which says that if the robot knows that he is not holding anything, that there is an object positioned at most k + 1 squares away along the row he is facing, and that there is no object where he is, then by doing FORWARD, he is able to achieve knowing that he is not holding anything and that there is an object positioned at most k squares in front of him.

Lemma 4.2

For all $\mathsf{k} \in \mathbb{N}$,
$\models \mathbf{Know}(\neg\mathsf{HS} \wedge \mathsf{OIF}(\mathsf{k}+1)) \wedge \mathbf{Know}(\neg\mathsf{OH}) \supset \mathbf{Can}(\textsc{forward}, \mathbf{Know}(\neg\mathsf{HS} \wedge \mathsf{OIF}(\mathsf{k})))$

Proof:
1. Let's assume that in some arbitrary structure,

$$\mathsf{e}, \mathsf{g} \models \mathbf{Know}(\neg\mathsf{HS} \wedge \mathsf{OIF}(\mathsf{k}+1)) \wedge \mathbf{Know}(\neg\mathsf{OH})$$

This means that for all e', such that $\langle \mathsf{e}, \mathsf{e}'\rangle \in \mathsf{K}$, $\mathsf{e}', \mathsf{g} \models \neg\mathsf{HS} \wedge \mathsf{OIF}(\mathsf{k}+1) \wedge \neg\mathsf{OH}$. Now we must show that $\mathsf{e}, \mathsf{g} \models \mathbf{Can}(\textsc{forward}, \mathbf{Know}(\neg\mathsf{HS} \wedge \mathsf{OIF}(\mathsf{k})))$. By definition 3.8, this amounts to showing that

$$\mathsf{e}, \mathsf{g} \models \exists d\, \mathbf{Know}(d = \textsc{forward} \wedge \mathbf{Res}(\textsc{forward}, \mathbf{Know}(\neg\mathsf{HS} \wedge \mathsf{OIF}(\mathsf{k}))))$$

that is, that there exists a d, such that for all $\mathsf{e}', \langle \mathsf{e}, \mathsf{e}'\rangle \in \mathsf{K}$,

$$\mathsf{e}', \mathsf{g_d} \models d = \textsc{forward} \wedge \mathbf{Res}(\textsc{forward}, \mathbf{Know}(\neg\mathsf{HS} \wedge \mathsf{OIF}(\mathsf{k})))$$

where $\mathsf{g_d} = \mathsf{g}\{d \to \mathsf{d}\}$.
2. Let's first show that there exists a d, such that for all $\mathsf{e}', \langle \mathsf{e}, \mathsf{e}'\rangle \in \mathsf{K}$, $\mathsf{e}', \mathsf{g_d} \models d = \textsc{forward}$. By assumption 4.2, we must have that $\mathsf{e}, \mathsf{g} \models \exists d\, \mathbf{Know}(d = \textsc{forward})$, and so the above must hold.

3. So what remains to be proven is that $e', g_d \models \mathbf{Res}(\textsc{forward}, \mathbf{Know}(\neg HS \wedge OIF(k)))$. By 1 and lemma 4.1, we must have that $e', g_d \models \mathbf{Res}(\textsc{forward}, \neg HS \wedge OIF(k))$. So by 2 and proposition 3.62, we have that $e', g_d \models \mathbf{Res}(d, \neg HS \wedge OIF(k))$. Now since K is transitive, we must then have that $e', g_d \models \mathbf{Know}(\mathbf{Res}(d, \neg HS \wedge OIF(k)))$. Then, by proposition 3.78, it follows that $e', g_d \models \mathbf{Res}(d, \mathbf{Know}(\neg HS \wedge OIF(k)))$. Finally, by 2 and proposition 3.62, we must have that $e', g_d \models \mathbf{Res}(\textsc{forward}, \mathbf{Know}(\neg HS \wedge OIF(k)))$. ■

Next, we must prove the following lemma about the SENSE action; it says that if the robot is not holding anything and there is an object positioned at most k squares in front of him, then after he does SENSE, it must still be the case that he is not holding anything and there is an object positioned at most k squares in front of him.

Lemma 4.3

For all $k \in \mathbb{N}$, $\models \neg HS \wedge OIF(k) \supset \mathbf{AfterNec}(\textsc{sense}, \neg HS \wedge OIF(k))$

Proof: From assumptions 4.16, 4.17, 4.13, and 4.18.

From this, the assumption specifying the effects of SENSE, and the one stating that all basic actions are known (assumptions 4.14 and 4.2), we can now prove the following lemma, which says that if the robot knows that he is not holding anything and that there is an object positioned at most k squares in front of him, then by doing SENSE, he is able to achieve knowing that he is not holding anything and that there is an object positioned at most k squares in front of him, and knowing whether there is an object where he is.

Lemma 4.4

For all $k \in \mathbb{N}$,

$\models \mathbf{Know}(\neg HS \wedge OIF(k)) \supset \mathbf{Can}(\textsc{sense}, \mathbf{Know}(\neg HS \wedge OIF(k)) \wedge \mathbf{Kwhether}(OH))$

Proof: From assumptions 4.2 and 4.14, and lemma 4.3.

Then, we can prove the following proposition about the scanning action; it says that if the robot knows that he is not holding anything and that there is an object positioned at most k squares in front of him and knows whether there is an object where he is, then he can achieve the goal of being where an object is by scanning forward up to k squares until he senses that an object is present.

Lemma 4.5

For all $k \in \mathbb{N}$, $\models \mathbf{Know}(\neg HS \wedge OIF(k)) \wedge \mathbf{Kwhether}(OH) \supset \mathbf{Can}(\textsc{scan}_k(OH), OH)$

Proof:

We prove this by induction over k.

Base case: We must show that

$$\models \mathbf{Know}(\neg\mathrm{HS} \wedge \mathrm{OIF}(0)) \wedge \mathbf{Kwhether}(\mathrm{OH}) \supset \mathbf{Can}(\textsc{scan}_0(\mathrm{OH}), \mathrm{OH})$$

It is easy to see that this must hold. $\textsc{scan}_0(\mathrm{OH}) \overset{\mathrm{def}}{=}$ **skip**, and $\mathbf{Can}(\mathbf{skip}, \mathrm{OH}) \overset{\mathrm{def}}{=} \mathbf{Know}(\mathrm{OH})$, so $\mathbf{Can}(\textsc{scan}_0(\mathrm{OH}), \mathrm{OH}) \overset{\mathrm{def}}{=} \mathbf{Know}(\mathrm{OH})$. Now by the facts of analytic geometry,

$$\models \mathrm{OIF}(0) \equiv \exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \langle 0, 0\rangle) \equiv \mathrm{OH}$$

so by propositions 3.17 and 3.11, we have that $\models \mathbf{Know}(\neg\mathrm{HS} \wedge \mathrm{OIF}(0)) \supset \mathbf{Know}(\mathrm{OH})$.

Induction step: We must show that if

$$\models \mathbf{Know}(\neg\mathrm{HS} \wedge \mathrm{OIF}(\mathsf{k})) \wedge \mathbf{Kwhether}(\mathrm{OH}) \supset \mathbf{Can}(\textsc{scan}_{\mathsf{k}}(\mathrm{OH}), \mathrm{OH})$$

then it must be the case that

$$\models \mathbf{Know}(\neg\mathrm{HS} \wedge \mathrm{OIF}(\mathsf{k}+1)) \wedge \mathbf{Kwhether}(\mathrm{OH}) \supset \mathbf{Can}(\textsc{scan}_{\mathsf{k}+1}(\mathrm{OH}), \mathrm{OH})$$

Now $\textsc{scan}_{\mathsf{k}+1}(\mathrm{OH}) \overset{\mathrm{def}}{=} \mathbf{if}(\neg\mathrm{OH}, \textsc{forward}; \textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{OH}), \mathbf{skip})$, and thus,

$$\begin{aligned}\mathbf{Can}(\textsc{scan}_{\mathsf{k}+1}(\mathrm{OH}), \mathrm{OH}) \overset{\mathrm{def}}{=}\ & \mathbf{Know}(\mathrm{OH}) \vee \\ & (\mathbf{Know}(\neg\mathrm{OH}) \wedge \mathbf{Can}(\textsc{forward}; \textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{OH}), \mathrm{OH}))\end{aligned}$$

The antecedent says that $\mathbf{Kwhether}(\mathrm{OH})$, that is, $\mathbf{Know}(\mathrm{OH}) \vee \mathbf{Know}(\neg\mathrm{OH})$, so all we need to prove is that

$$\models \mathbf{Know}(\neg\mathrm{HS} \wedge \mathrm{OIF}(\mathsf{k}+1)) \wedge \mathbf{Know}(\neg\mathrm{OH}) \supset \mathbf{Can}(\textsc{forward}; \textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{OH}), \mathrm{OH})$$

From lemma 4.4 and the induction hypothesis, it follows by proposition 3.80 that

$$\models \mathbf{Know}(\neg\mathrm{HS} \wedge \mathrm{OIF}(\mathsf{k})) \supset \mathbf{Can}(\textsc{sense}; \textsc{scan}_{\mathsf{k}}, \mathrm{OH})$$

From this and lemma 4.2, the desired result follows by proposition 3.80. ∎

From this, we easily obtain the desired result, that is, that if the robot knows that there is an object that is positioned at most k squares away along the row he is facing and that he is not holding anything, then he can get where some object is by sensing and then scanning forward, up to k squares, until he senses that an object is present.

Proposition 4.11

For all $\mathsf{k} \in \mathbb{N}$, $\models \mathbf{Know}(\neg\mathrm{HS} \wedge \mathrm{OIF}(\mathsf{k})) \supset \mathbf{Can}((\textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{OH})), \mathrm{OH})$

Proof:

This follows trivially from lemmas 4.5 and 4.4 by proposition 3.80. ∎

So it is quite clear that ability to act upon an object does not require knowing its relative position. But then what is required? It seems that the best we can say is that the agent must *know of some procedure* that will take him to where the object is.

But this creates problems in the formalization of ability to do certain high-level parametrized actions, for example, the action of "going to the position of an object θ" $\text{GOWHERE}(\theta)$. It would be inappropriate to treat this action as a primitive because we want to model how knowledge enables action at a more detailed level. The other way to way to deal with such an action within our (and Moore's) framework would involve defining it in terms of lower-level actions that are parametrized with the information that must actually be known in order to be able to do the high-level action (recall Moore's proposal for the action of "putting a block on another"). This allows knowledge prerequisites to be enforced by the requirement that one know which primitive action to do next and removes the need to formalize them explicitly. But for actions like $\text{GOWHERE}(\theta)$, it is not clear how this could be put into practice.

However, notice that $\text{GOWHERE}(\theta)$ is a strange kind of action in that it appears to refer to anything that would achieve the goal that the agent be where θ is; it is as much like a *goal* as like an action. Perhaps we should rule out the introduction of such actions, but instead provide an action-less version of the **Can** operator: $\mathbf{CanAch}(\varphi)$ would mean that **self** is able to achieve φ in one way or another. Then, we may use $\mathbf{CanAch}(\text{POS}(\theta) = \mathbf{here} \wedge \varphi)$ instead of something like $\mathbf{Can}(\text{GOWHERE}(\theta), \varphi)$.[8] A coarse "syntactic" way of formalizing **CanAch** goes as follows: $\mathbf{e}, \mathbf{g} \models \mathbf{CanAch}(\varphi)$ iff there exists an action expression δ such that $\mathbf{e}, \mathbf{g} \models \mathbf{Can}(\delta, \varphi)$. A more general and robust approach is being developed by Nunes (1990).

4.8 Map Navigation

Before moving on, let us examine one last area of robotics, that of navigation with the help of a map. In order to fully take advantage of the information contained in a map, say to find out how to get to a destination, an agent must first orient himself with respect to it, that is, find out where he is on the map, what his absolute position is. If he does not already know this, he must try to match the landmarks represented on the map with features of his current environment. Our simple domain provides instances of this if we treat objects of various shapes as landmarks. Consider a variant of the scanning example of the previous

[8]This assumes that it is known that θ refers to the same entity before and after the action is done; the assumption can be dispensed with by referring to the denotation of θ prior to the action as illustrated earlier.

section, where the robot knows what the absolute position of the unique object of shape s is and knows that this object is positioned at most k squares in front of him; the robot should be able get into a state where he knows where he is by scanning forward until he detects that object. The following proposition shows that this is essentially correct. It says that if the robot knows that the unique object with shape s is at absolute position $\vec{p}$, that it is positioned at most k squares away along the row he is facing, and that he is not holding anything, then he can achieve the goal of knowing that he is at absolute position $\vec{p}$ by sensing and then scanning forward, up to k squares, until he senses that an object with shape s is present.[9]

Proposition 4.12

For all $\mathsf{k} \in \mathbb{N}$,

$$\begin{aligned}\models \forall \vec{p} \forall s(\mathbf{Know}(&\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \vec{p} \wedge \forall y(\textsc{OfShape}(y,s) \equiv y = x) \wedge \\ &\exists n(\textsc{rpos}(x) = \langle n, 0\rangle \wedge 0 \leq n \leq \mathsf{k})) \wedge \neg\exists y \textsc{Holding}(y)) \supset \\ &\mathbf{Can}((\textsc{sense}; \textsc{scan}_{\mathsf{k}}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here} \wedge \textsc{OfShape}(x,s)))), \\ &\quad \mathbf{Know}(\mathbf{here} = \vec{p})))\end{aligned}$$

We prove this proposition in a similar manner to proposition 4.11. First, let us introduce the following abbreviations:

Definition 4.11

$\mathrm{US}(\theta^i_x, \theta^i_s) \overset{\text{def}}{=} \forall v^i(\textsc{OfShape}(v^i, \theta^i_s) \equiv v^i = \theta^i_x)$, provided v^i does not occur free in θ^i_x and θ^i_s

$\mathrm{OHS}(\theta^i_s) \overset{\text{def}}{=} \exists v^i(\textsc{Object}(v^i) \wedge \textsc{pos}(v^i) = \mathbf{here} \wedge \textsc{OfShape}(v^i, \theta^i_s)$ provided v^i does not occur free in θ^i_s

$\mathrm{OSPIF}(\theta^i_s, \theta^i_p, \mathsf{k}) \overset{\text{def}}{=}$

$\exists v^i_x(\textsc{Object}(v^i_x) \wedge \textsc{pos}(v^i_x) = \theta^i_p \wedge \mathrm{US}(v^i_x, \theta^i_s) \wedge \exists v^i_n(\textsc{rpos}(v^i_x) = \langle v^i_n, 0\rangle \wedge 0 \leq v^i_n \leq \mathsf{k}))$,
where $\mathsf{k} \in \mathbb{N}$, provided v^i_x and v^i_n do not occur free in θ^i_s and θ^i_p

We first extend the result of lemma 4.1 about action FORWARD into the following lemma, which says that if the unique object with shape s is at absolute position $\vec{p}$ and is positioned

[9] In this case, the requirement that the agent know that he is not holding anything is necessary, because he must sense the shape of whatever object he encounters, and if he were holding something, his sensors might get confused between the two objects (see assumption 4.15).

at most $\mathsf{k}+1$ squares in front of the agent, and he is not holding anything, and there is no object of shape s here he is, then his doing FORWARD must result in the unique object with shape s being at absolute position $\vec{p}$ and being positioned at most k squares in front of him, and his not holding anything.

Lemma 4.6

For all $\mathsf{k} \in \mathbb{N}$,
$\models \forall\vec{p}\forall s(\mathrm{OSPIF}(s,\vec{p},\mathsf{k}+1) \wedge \neg\mathrm{HS} \wedge \neg\mathrm{OHS} \supset \mathbf{Res}(\textsc{forward}, \mathrm{OSPIF}(s,\vec{p},\mathsf{k}) \wedge \neg\mathrm{HS}))$

Proof: From lemma 4.1, and assumptions 4.6 and 4.19.

From this, we can then prove the following ability result about FORWARD; it says that if the robot knows that the unique object with shape s is at absolute position $\vec{p}$ and is positioned at most $\mathsf{k}+1$ squares in front of him, and that he is not holding anything, and knows that there is no object of shape s where he is, then by doing FORWARD, he is able to achieve knowing that the unique object with shape s is at absolute position $\vec{p}$ and is positioned at most k squares in front of him, and that he is not holding anything.

Lemma 4.7

For all $\mathsf{k} \in \mathbb{N}$,
$$\models \forall\vec{p}\forall s(\mathbf{Know}(\mathrm{OSPIF}(s,\vec{p},\mathsf{k}+1) \wedge \neg\mathrm{HS}) \wedge \mathbf{Know}(\neg\mathrm{OH}) \supset \mathbf{Can}(\textsc{forward}, \mathbf{Know}(\mathrm{OSPIF}(s,\vec{p},\mathsf{k}) \wedge \neg\mathrm{HS})))$$

Proof:
Same argument as for lemma 4.2, except that lemma 4.6 instead of lemma 4.1 is used at step 3. ■

Next, we must extend the result of lemma 4.3 about action SENSE. This yields the following lemma, which says that if the unique object with shape s is at absolute position $\vec{p}$ and is positioned at most k squares in front of the agent, and he is not holding anything, then his doing SENSE must result in the unique object with shape s being at absolute position $\vec{p}$ and being positioned at most k squares in front of him, and his not holding anything.

Lemma 4.8

For all $\mathsf{k} \in \mathbb{N}$, $\models \forall\vec{p}\forall s(\mathrm{OSPIF}(s,\vec{p},\mathsf{k}) \wedge \neg\mathrm{HS} \supset \mathbf{AfterNec}(\textsc{sense}, \mathrm{OSPIF}(s,\vec{p},\mathsf{k}) \wedge \neg\mathrm{HS}))$

Proof: From lemma 4.3, and assumptions 4.16 and 4.19.

From this, we can then prove the following ability result about SENSE; it says that if the robot knows that the unique object with shape s is at absolute position $\vec{p}$ and is positioned at most k squares in front of him, and that he is not holding anything, then by doing SENSE, he is able to achieve knowing that the unique object with shape s is at absolute position $\vec{p}$ and is positioned at most k squares in front of him, and that he is not holding anything, and knowing whether there is an object of shape s where he is.

Lemma 4.9

For all $\mathsf{k} \in \mathbb{N}$,

$$\models \forall\vec{p}\forall s(\mathbf{Know}(\mathrm{OSPIF}(s,\vec{p},\mathsf{k}) \wedge \neg\mathrm{HS}) \supset \mathbf{Can}(\textsc{sense}, \mathbf{Know}(\mathrm{OSPIF}(s,\vec{p},\mathsf{k}) \wedge \neg\mathrm{HS}) \wedge \mathbf{Kwhether}(\mathrm{OHS}(s))))$$

Proof:
Same argument as for lemma 4.4, except that lemma 4.8 instead of lemma 4.3 and assumption 4.15 instead of assumption 4.14 are used at step 3. ■

Then, we can prove the following proposition about the scanning action; it says that if the robot knows that the unique object with shape s is at absolute position $\vec{p}$, that it is positioned at most k squares away along the row he is facing, and that he is not holding anything, and knows whether there is an object of shape s where he is, then he can achieve the goal of knowing that he is at absolute position $\vec{p}$ by scanning forward, up to k squares, until he senses that an object with shape s is present.

Lemma 4.10

For all $\mathsf{k} \in \mathbb{N}$,

$$\models \forall\vec{p}\forall s(\mathbf{Know}(\mathrm{OSPIF}(s,\vec{p},\mathsf{k}) \wedge \neg\mathrm{HS}) \wedge \mathbf{Kwhether}(\mathrm{OHS}(s)) \supset \mathbf{Can}(\textsc{scan}_{\mathsf{k}}(\mathrm{OHS}(s)), \mathbf{Know}(\mathrm{here} = \vec{p})))$$

Proof:
We prove this by induction over k.
Base case: We must show that

$$\models \forall\vec{p}\forall s(\mathbf{Know}(\mathrm{OSPIF}(s,\vec{p},0) \wedge \neg\mathrm{HS}) \wedge \mathbf{Kwhether}(\mathrm{OHS}(s)) \supset \mathbf{Can}(\textsc{scan}_0(\mathrm{OHS}(s)), \mathbf{Know}(\mathrm{here} = \vec{p})))$$

It is easy to see that this must hold. $\textsc{scan}_0(\varphi) \overset{\text{def}}{=} \textbf{skip}$, and $\textbf{Can}(\textbf{skip},\varphi) \overset{\text{def}}{=} \textbf{Know}(\varphi)$, so $\textbf{Can}(\textsc{scan}_0(\text{OHS}(s)),\textbf{Know}(\textbf{here} = \vec{p}))) \overset{\text{def}}{=} \textbf{Know}(\textbf{Know}(\textbf{here} = \vec{p}))$. Now by the facts of analytic geometry,

$$\models \text{OSPIF}(s,\vec{p},0) \equiv \exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \vec{p} \wedge \text{US}(x,s) \wedge \textsc{rpos}(x) = \langle 0,0\rangle)$$

and thus, $\models \text{OSPIF}(s,\vec{p},0) \supset \textbf{here} = \vec{p}$. Thus by propositions 3.17, 3.11, and 3.14, we have that $\models \textbf{Know}(\text{OSPIF}(s,\vec{p},0) \wedge \neg\text{HS}) \supset \textbf{Know}(\textbf{Know}(\textbf{here} = \vec{p}))$.
Induction step: We must show that if

$$\begin{aligned}&\models \textbf{Know}(\text{OSPIF}(s,\vec{p},\mathsf{k}) \wedge \neg\text{HS}) \wedge \textbf{Kwhether}(\text{OHS}(s)) \supset\\ &\quad \textbf{Can}(\textsc{scan}_\mathsf{k}(\text{OHS}(s)),\textbf{Know}(\textbf{here} = \vec{p}))\end{aligned}$$

then it must be the case that

$$\begin{aligned}&\models \textbf{Know}(\text{OSPIF}(s,\vec{p},\mathsf{k}+1) \wedge \neg\text{HS}) \wedge \textbf{Kwhether}(\text{OHS}(s)) \supset\\ &\quad \textbf{Can}(\textsc{scan}_{\mathsf{k}+1}(\text{OHS}(s)),\textbf{Know}(\textbf{here} = \vec{p}))\end{aligned}$$

Now $\textsc{scan}_{\mathsf{k}+1}(\varphi) \overset{\text{def}}{=} \textbf{if}(\neg\varphi, \textsc{forward}; \textsc{sense}; \textsc{scan}_\mathsf{k}(\varphi), \textbf{skip})$, and thus,

$$\begin{aligned}&\textbf{Can}(\textsc{scan}_{\mathsf{k}+1}(\text{OHS}(s)),\textbf{Know}(\textbf{here} = \vec{p})) \overset{\text{def}}{=}\\ &\quad (\textbf{Know}(\text{OHS}(s)) \wedge \textbf{Know}(\textbf{here} = \vec{p})) \vee\\ &\quad (\textbf{Know}(\neg\text{OHS}(s)) \wedge \textbf{Can}(\textsc{forward}; \textsc{sense}; \textsc{scan}_\mathsf{k}(\text{OHS}(s)),\textbf{Know}(\textbf{here} = \vec{p})))\end{aligned}$$

The antecedent says that $\textbf{Kwhether}(\text{OHS}(s))$, that is, $\textbf{Know}(\text{OHS}(s)) \vee \textbf{Know}(\neg\text{OHS}(s))$. So let's consider each alternative in turn. In the former case, we need to prove that

$$\models \textbf{Know}(\text{OSPIF}(s,\vec{p},\mathsf{k}+1) \wedge \neg\text{HS}) \wedge \textbf{Know}(\text{OHS}(s)) \supset \textbf{Know}(\textbf{here} = \vec{p})$$

Since $\models \text{OSPIF}(s,\vec{p},\mathsf{k}+1) \wedge \text{OHS}(s) \supset \textbf{here} = \vec{p}$, the above must hold by propositions 3.17 and 3.11. In the latter case, we must prove that

$$\begin{aligned}&\models \textbf{Know}(\text{OSPIF}(s,\vec{p},\mathsf{k}+1) \wedge \neg\text{HS}) \wedge \textbf{Know}(\neg\text{OHS}(s)) \supset\\ &\quad \textbf{Can}(\textsc{forward}; \textsc{sense}; \textsc{scan}_\mathsf{k}(\text{OHS}(s)),\textbf{Know}(\textbf{here} = \vec{p}))\end{aligned}$$

From lemma 4.9 and the induction hypothesis, it follows by proposition 3.80 that

$$\models \textbf{Know}(\text{OSPIF}(s,\vec{p},\mathsf{k}) \wedge \neg\text{HS}) \supset \textbf{Can}(\textsc{sense}; \textsc{scan}_\mathsf{k}(\text{OHS}(s)),\textbf{Know}(\textbf{here} = \vec{p}))$$

From this and lemma 4.7, the desired result follows by proposition 3.80. ■

From this, it is easy to show that the desired result holds, that is, that if the robot knows that the unique object with shape s is at absolute position $\vec{p}$, that it is positioned at most k squares away along the row he is facing, and that he is not holding anything, then he can achieve the goal of knowing that he is at absolute position $\vec{p}$ by sensing, and then scanning forward, up to k squares, until he senses that an object with shape s is present.

Proposition 4.12

$$\begin{aligned}&\text{For all } \mathsf{k} \in \mathbb{N},\\ &\models \forall \vec{p} \forall s(\mathbf{Know}(\mathrm{OSPIF}(s, \vec{p}, \mathsf{k}) \land \neg \mathrm{HS}) \supset \\ &\qquad \mathbf{Can}((\textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{OHS}(s))), \mathbf{Know}(\mathbf{here} = \vec{p})))\end{aligned}$$

Proof:
This follows trivially from lemmas 4.10 and 4.9 by proposition 3.80. ■

We could also show that an agent can find out what his absolute orientation is by recognizing objects that have a known orientation with respect to each other. And it is clear that once an agent knows his absolute position, he can use the map to navigate to some object represented on it. These map navigation examples exploit a key feature of our framework, which is that it allows both indexical and absolute knowledge to be represented, and relations between the two to be expressed (this feature of the map navigation problem was pointed out by Israel (1987)). This distinguishes it from the indexical version of the situation calculus proposed by Subramanian and Woodfill (1989), where one simply introduces indexical entities in the ontology.

Chapter 5

Other Applications

The need for the notion of indexical knowledge in the formalization of the prerequisites and effects of actions is most easily motivated by looking at actions in the robotics domain — actions taking place in physical space — as in the previous chapter. The notion itself is most clearly understood in terms of such instances, where the agent has knowledge that is relative to his point of view in physical space. This might suggest that the notion is bound to that of physical space. But this is not the case. The notion is really rather abstract. It appears useful as long as the domain involves agents that operate in some kind of space, from some kind of point of view into that space.

In this chapter we provide evidence for this claim by formalizing various non-robotics applications, some of which involve very abstract notions of space. The phone system can be viewed as one such kind of space and an agent can be quite ignorant of what characterizes his absolute position in that space, things like area-code, country code, etc. In section 5.1, we formalize an example that reflects this indexicality. In the following section, we look at the domain of data structure search and manipulation. At first, it might seem strange to look at this in terms of an agent located in some kind of space and having knowledge about that space that is relative to his point of view, yet this seems very much to be the outlook taken by many algorithms; one talks about following pointers and searching graphs. We show that our framework has good claims of applicability to this domain. Following this in section 5.3, we look at three examples where temporal knowledge is required for ability. There is as much of a need for keeping track of one's place in time as of one's place in space. We find that the distinctions between various kinds of knowledge that proved useful in spatial contexts like the robotics situations of chapter 4 (i.e., knowing the absolute position of something, knowing the relative position of something, and knowing how to get where something is) are similarly useful in dealing with ability in temporal contexts. We also find that our framework provides the necessary tools for producing a reasonable formalization of these temporal examples. In the section 5.4, we discuss problems with the notion of *de re* knowledge and its role in action. In the last section, we review the main points that the examples of this chapter are trying to make. We also argue that the distinction between indexical and objective knowledge supported by our framework has substantial practical value and that it cannot be done justice within existing accounts of ability. Note that complete proofs are supplied only for the example in section 5.1; for the others, we only sketch how the proofs would go.

5.1 Making a Phone Call

Our first application is concerned with the ability of an agent to make a phone call. This involves an action that is rather abstract, in comparison with the robotics actions discussed earlier. We will take calling to be a procedure that takes a destination phone as parameter; it requires the presence of a source phone at the agent's current position, and involves dialing on the source phone the sequence of digits required for a connection to be established between it and the destination phone. We will represent an instance of the procedure being applied to a destination phone denoted by θ^i_{pd} by the expression $\textsc{call}(\theta^i_{pd})$. If we were to develop a formalization that takes this as a primitive, we would end up requiring an agent to know which phone θ^i_{pd} is in order to be judged able to establish a connection to it by doing the action. It is far from obvious what having *de re* knowledge of a phone should amount to: perhaps knowing the complete number of the phone including area code, perhaps knowing its manufacturer and serial number, or perhaps knowing its absolute position. But no matter which interpretation we take, this requirement is unnecessary. This is obvious in the latter two cases. In the former case: if the agent knows that the call is local, that is, that his own area code is the same as that of the phone he wants to call, then he does not need to know what that area code is in order to be able to make the call.

We will now develop a solution that captures more adequately the knowledge prerequisites of the action. Our approach involves defining CALL in terms of less abstract actions that take more appropriate — more indexical — parameters. For simplicity, we will assume that calls belong to one of two classes: local calls, which require only the number of the destination phone to be dialed, and long-distance calls, which also require its area code to be dialed (thus, we ignore international calls, non-local calls within the same area, etc.). CALL is defined as a conditional action that selects between the actions DIALLOC and DIALLD depending on which class the call belongs to. To make a local call, one does action DIALLOC, which takes as parameter the number of the destination phone; the action involves dialing this number on a source phone that should be positioned where the agent is. For a long-distance call, one does action DIALLD, which takes an additional parameter, the area code of the destination phone; the action involves dialing the appropriate prefix for long-distance, the area code, and the number on a source phone that is at the agent's current position. Formally, this yields the following:

Definition 5.1.1 (Action CALL)

$$\textsc{call}(\theta^i_{pd}) \overset{\text{def}}{=}$$
$$\textbf{if}(\textsc{ac}(\textbf{here}) = \textsc{ac}(\textsc{pos}(\theta^i_{pd})), \textsc{dialloc}(\textsc{no}(\theta^i_{pd})), \textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta^i_{pd})), \textsc{no}(\theta^i_{pd})))$$

This says that that calling a destination phone θ^i_{pd} (on a source phone that is **here**) amounts to doing a local dialing of the number of θ^i_{pd} if the area codes of the positions of the agent and θ^i_{pd} are the same, and doing a long-distance dialing of the area code of the position of θ^i_{pd} and number of θ^i_{pd} otherwise.[1]

We will be assuming that the agent knows how to do the procedures DIALLOC and DAILLD, that is technically, that they are epistemically rigid functions. Given this, the way CALL is defined has the following consequences:

- the agent need not know which phone he is calling; if the call is long-distance, knowing its area code and number is sufficient, and if the call is local, knowing its number is sufficient;

- the agent need not know which phone he is calling from, since the source phone is not a parameter; whatever phone happens to be where he is positioned is used.

[1]In reality, the source phone need not be exactly where the agent is, it need only be within reach; but our treatment of this action involves the same simplifications as were present in the treatment of manipulation actions in chapter 4.

This is much more realistic as a characterization of the knowledge prerequisites of CALL than what we had with the first approach outlined.

We will soon prove that if these knowledge prerequisites are satisfied, then the agent is able to make a phone call. But first, we must complete our formalization of the domain. The effects of the action DIALLOC are specified by the following assumption:

Assumption 5.1.1 (Effects of DIALLOC)

$$\begin{aligned}\models \forall p_s \forall p_d(&\text{PHONE}(p_s) \wedge \text{POS}(p_s) = \textbf{here} \wedge \neg\text{INUSE}(p_s) \wedge \\ &\text{PHONE}(p_d) \wedge \text{AC}(\textbf{here}) = \text{AC}(\text{POS}(p_d)) \supset \\ &\textbf{Res}(\text{DIALLOC}(\text{NO}(p_d)), \text{CONNECTED}(p_s, p_d)))\end{aligned}$$

This says that if p_s is an unused phone positioned where the agent is and the area code of the absolute position of a phone p_d is the same as that of the agent, then the agent's doing a local dialing of the number of p_d (on the phone that is **here**) results in a connection being established between the two phones.

We also assume that the agent knows how to accomplish the DIALLOC procedure, that is, that DIALLOC is an epistemically rigid function. We specify this as follows:

Assumption 5.1.2 (DIALLOC is epistemically rigid)

$$\models \forall n(\textbf{Know}(\text{ISPHNO}(n)) \supset \exists d\, \textbf{Know}(d = \text{DIALLOC}(n)))$$

This says that the agent knows what the action of making a local dialing of a phone number n is if he knows what n is.

We make similar assumptions concerning the effects and epistemic rigidity of the action of making a long-distance dialing DIALLD. Note that ISAC(θ^i) means that θ^i is an area code.

Assumption 5.1.3 (Effects of DIALLD)

$$\begin{aligned}\models \forall p_s \forall p_d(&\text{PHONE}(p_s) \wedge \text{POS}(p_s) = \textbf{here} \wedge \neg\text{INUSE}(p_s) \wedge \\ &\text{PHONE}(p_d) \wedge \text{AC}(\textbf{here}) \neq \text{AC}(\text{POS}(p_d)) \supset \\ &\textbf{Res}(\text{DIALLD}(\text{AC}(\text{POS}(p_d)), \text{NO}(p_d)), \text{CONNECTED}(p_s, p_d)))\end{aligned}$$

Assumption 5.1.4 (DIALLD **is epistemically rigid)**

$$\models \forall c \forall n(\mathbf{Know}(\textsc{IsPhNo}(n) \wedge \textsc{IsAc}(c)) \supset \exists d\, \mathbf{Know}(d = \textsc{dialld}(c, n)))$$

We need a few additional assumptions about the domain. Assumption 5.1.5 says that n is a phone number if it is the number of some phone and assumption 5.1.6 says that c is an area code if it is the area code of the position of something.

Assumption 5.1.5 $\models \forall p \forall n(\textsc{Phone}(p) \wedge n = \textsc{no}(p) \supset \textsc{IsPhNo}(n))$

Assumption 5.1.6 $\models \forall c \forall x(c = \textsc{ac}(x) \supset \textsc{IsAc}(c))$

Now that we have developed a suitable formalization of the domain, we are ready to prove some results concerning the agent's ability to make phone calls. Our main objective is to prove that if the agent's knowledge satisfies certain requirements, he must then be able to achieve the goal of establishing a connection between two phones by doing the action CALL. Now to show ability to achieve a conditional action such as CALL, one must show that either the agent knows that the test condition holds and is able to achieve the goal by doing the "then" branch, or knows that the condition does not hold and is able to achieve the goal by doing the "else" branch. Thus before proving ability to achieve the goal by doing CALL, we will prove ability results involving its sub-actions DIALLOC and DIALLD.

We first prove proposition 5.1.1, which says that if the agent knows that there is an unused phone positioned where he is, and knows what the number of a destination phone θ_{pd} is, and knows that his own area code is the same as that of θ_{pd}, then he can achieve the goal of establishing a connection between some two phones by doing a local dialing of the number of θ_{pd} (on the phone that is **here**):

Proposition 5.1.1

$$\begin{aligned} \models \exists n\, \mathbf{Know}(&\exists p_s(\textsc{Phone}(p_s) \wedge \textsc{pos}(p_s) = \mathbf{here} \wedge \neg\textsc{InUse}(p_s)) \wedge \\ &\textsc{Phone}(\theta_{pd}) \wedge n = \textsc{no}(\theta_{pd}) \wedge \textsc{ac}(\mathbf{here}) = \textsc{ac}(\textsc{pos}(\theta_{pd}))) \supset \\ &\mathbf{Can}(\textsc{dialloc}(\textsc{no}(\theta_{pd})), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d)) \end{aligned}$$

Proof:
1. Assume that

$$\begin{aligned} e, g \models \exists n\, \mathbf{Know}(&\exists p_s(\textsc{Phone}(p_s) \wedge \textsc{pos}(p_s) = \mathbf{here} \wedge \neg\textsc{InUse}(p_s)) \wedge \\ &\textsc{Phone}(\theta_{pd}) \wedge n = \textsc{no}(\theta_{pd}) \wedge \textsc{ac}(\mathbf{here}) = \textsc{ac}(\textsc{pos}(\theta_{pd}))) \end{aligned}$$

that is, that there exists n, such that for all $e' = \langle w', a', t' \rangle$, $\langle e, e' \rangle \in K$,

$$\begin{aligned} e', g_n \models & \exists p_s(\text{PHONE}(p_s) \land \text{POS}(p_s) = \textbf{here} \land \neg\text{INUSE}(p_s)) \land \\ & \text{PHONE}(\theta_{pd}) \land n = \text{NO}(\theta_{pd}) \land \text{AC}(\textbf{here}) = \text{AC}(\text{POS}(\theta_{pd})) \end{aligned}$$

where $g_n = g\{n \to n\}$. We must show that

$$e, g \models \textbf{Can}(\text{DIALLOC}(\text{NO}(\theta_{pd})), \exists p_s \exists p_d\, \text{CONNECTED}(p_s, p_d))$$

By the definition of **Can**, this means that

$$\begin{aligned} e, g \models \exists d\, \textbf{Know}(& d = \text{DIALLOC}(\text{NO}(\theta_{pd})) \land \\ & \textbf{Res}(\text{DIALLOC}(\text{NO}(\theta_{pd})), \exists p_s \exists p_d\, \text{CONNECTED}(p_s, p_d))) \end{aligned}$$

that is, that there exists a d, such that for all e', $\langle e, e' \rangle \in K$,

$$\begin{aligned} e', g_d \models & d = \text{DIALLOC}(\text{NO}(\theta_{pd})) \land \\ & \textbf{Res}(\text{DIALLOC}(\text{NO}(\theta_{pd})), \exists p_s \exists p_d\, \text{CONNECTED}(p_s, p_d)) \end{aligned}$$

where $g_d = g\{d \to d\}$.
2. Let's show that there exists a d, such that for all e', such that $\langle e, e' \rangle \in K$, $e', g_d \models d = \text{DIALLOC}(\text{NO}(\theta_{pd}))$. By 1 and assumption 5.1.5, we must have that $e, g_n \models \text{ISPHNO}(n)$. So by 1 and assumption 5.1.2, we must have that $e, g_n \models \exists d\, \textbf{Know}(d = \text{DIALLOC}(n))$. Since by 1, we have that $e', g_n \models n = \text{NO}(\theta_{pd})$, the above must also hold.
3. It remains to be proven that

$$e', g_d \models \textbf{Res}(\text{DIALLOC}(\text{NO}(\theta_{pd})), \exists p_s \exists p_d\, \text{CONNECTED}(p_s, p_d))$$

By assumption 5.1.1 and proposition 3.69, we have that

$$\begin{aligned} \models & \exists p_s(\text{PHONE}(p_s) \land \text{POS}(p_s) = \textbf{here} \land \neg\text{INUSE}(p_s)) \land \\ & \text{PHONE}(p_d) \land \text{AC}(\textbf{here}) = \text{AC}(\text{POS}(p_d)) \supset \\ & \textbf{Res}(\text{DIALLOC}(\text{NO}(p_d)), \exists p_s \exists p_d\, \text{CONNECTED}(p_s, p_d)) \end{aligned}$$

From this and proposition 3.62, we obtain that

$$\begin{aligned} e', g_n \models & \exists p_s(\text{PHONE}(p_s) \land \text{POS}(p_s) = \textbf{here} \land \neg\text{INUSE}(p_s)) \land \\ & \text{PHONE}(\theta_{pd}) \land \text{AC}(\text{POS}(\theta_{ps})) = \text{AC}(\text{POS}(\theta_{pd})) \supset \\ & \textbf{Res}(\text{DIALLOC}(\text{NO}(\theta_{pd})), \exists p_s \exists p_d\, \text{CONNECTED}(p_s, p_d)) \end{aligned}$$

From this and 1, the desired result follows. ■

Note that the agent is required only to have *de re* knowledge of the number of the destination phone; he need not have *de re* knowledge of either the destination or source phone. Note also how no requirement is made that the agent know who he is; all the knowledge that the agent is required to have about himself is indexical rather than absolute.

The goal specified in this proposition is not as strong as one might want, in that it does not require the actual phones involved in the call to be connected. One could strengthen the goal in various ways without requiring the agent to know which phones were involved. For example, one could establish that the phone with the number dialed was connected to the one that is positioned where the agent is, assuming that dialing does not change the number or position of phones. Or it could be established that the phone denoted by θ_{pd} before the action is now connected to the one that was positioned where the agent was before the action (assuming that the action does not change the denotation of θ_{pd}). This issue was discussed in chapter 4 and the suggestions given there are applicable.

We next prove a similar proposition for the long distance case (action DIALLD). The knowledge required is the same as in the DIALLOC case except that the agent must now know that his area code is different from that of the destination phone and know what the area code of this phone is.

Proposition 5.1.2

$$\begin{aligned}&\models \exists n \exists c\, \mathbf{Know}(\exists p_s(\textsc{Phone}(p_s) \land \textsc{pos}(p_s) = \mathbf{here} \land \neg\textsc{Inuse}(p_s)) \land \\ &\qquad\qquad \textsc{Phone}(\theta_{pd}) \land n = \textsc{no}(\theta_{pd}) \land c = \textsc{ac}(\textsc{pos}(\theta_{pd})) \land \textsc{ac}(\mathbf{here}) \neq \textsc{ac}(\textsc{pos}(\theta_{pd}))) \\ &\quad \supset \mathbf{Can}(\textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{no}(\theta_{pd})), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d))\end{aligned}$$

Proof:
Since the proof of this proposition is very similar to that of proposition 5.1.1, we do not state it fully; we describe only how it differs from the proof of proposition 5.1.1.
1. We assume that there exist n, and c, such that for all $\mathsf{e}' = \langle \mathsf{w}', \mathsf{a}', \mathsf{t}' \rangle$, $\langle \mathsf{e}, \mathsf{e}' \rangle \in \mathsf{K}$,

$$\begin{aligned}\mathsf{e}', \mathsf{g}_{\mathsf{nc}} &\models \exists p_s(\textsc{Phone}(p_s) \land \textsc{pos}(p_s) = \mathbf{here} \land \neg\textsc{Inuse}(p_s)) \land \\ &\quad \textsc{Phone}(\theta_{pd}) \land n = \textsc{no}(\theta_{pd}) \land c = \textsc{ac}(\textsc{pos}(\theta_{pd})) \land \textsc{ac}(\mathbf{here}) \neq \textsc{ac}(\textsc{pos}(\theta_{pd}))\end{aligned}$$

where $\mathsf{g}_{\mathsf{nc}} = \mathsf{g}\{n \to \mathsf{n}, c \to \mathsf{c}\}$. We must show that there exists a d, such that for all $\mathsf{e}', \langle \mathsf{e}, \mathsf{e}' \rangle \in \mathsf{K}$,

$$\begin{aligned}\mathsf{e}', \mathsf{g}_{\mathsf{d}} &\models d = \textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{no}(\theta_{pd})) \land \\ &\quad \mathbf{Res}(\textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{no}(\theta_{pd})), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d))\end{aligned}$$

where $\mathsf{g}_{\mathsf{d}} = \mathsf{g}\{d \to \mathsf{d}\}$.
2. We show that there exists a d, such that for all e', such that $\langle \mathsf{e}, \mathsf{e}' \rangle \in \mathsf{K}$, $\mathsf{e}', \mathsf{g}_{\mathsf{d}} \models d = \textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{no}(\theta_{pd}))$. By 1 and assumption 5.1.5, we must have that $\mathsf{e}, \mathsf{g}_{\mathsf{nc}} \models \textsc{IsPhNo}(n)$. By 1 and assumption 5.1.6, we must have that $\mathsf{e}, \mathsf{g}_{\mathsf{nc}} \models \textsc{IsAc}(c)$. So by 1 and assumption 5.1.4, we must have that $\mathsf{e}, \mathsf{g}_{\mathsf{nc}} \models \exists d\, \mathbf{Know}(d = \textsc{dialld}(c, n))$. Since by 1, we have that $\mathsf{e}', \mathsf{g}_{\mathsf{nc}} \models c = \textsc{ac}(\textsc{pos}(\theta_{pd})) \land n = \textsc{no}(\theta_{pd})$, the above must also hold.
3. What remains to be proven is that

$$\mathsf{e}', \mathsf{g}_{\mathsf{d}} \models \mathbf{Res}(\textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{no}(\theta_{pd})), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d))$$

By assumption 5.1.3 and propositions 3.69 and 3.62, we obtain that

$$\begin{aligned} e', g_{nc} \models & \exists p_s(\textsc{Phone}(p_s) \wedge \textsc{pos}(p_s) = \textbf{here} \wedge \neg\textsc{Inuse}(p_s)) \wedge \\ & \textsc{Phone}(\theta_{pd}) \wedge \textsc{ac}(\textbf{here}) \neq \textsc{ac}(\textsc{pos}(\theta_{pd})) \supset \\ & \textbf{Res}(\textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{no}(\theta_{pd})), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d)) \end{aligned}$$

By this and 1, the desired result follows. ∎

Finally, given these results, we can now prove that the agent can achieve the goal of establishing a connection between some two phones by calling phone θ_{pd}, if he knows that an unused phone is positioned where he is, knows what the number of θ_{pd} is, and either knows that his own area code is the same as that of θ_{pd}, or knows that they are different and knows what the area code of θ_{pd} is:

Proposition 5.1.3

$$\begin{aligned} & \models \exists n \textbf{Know}(\exists p_s(\textsc{Phone}(p_s) \wedge \textsc{pos}(p_s) = \textbf{here} \wedge \neg\textsc{Inuse}(p_s)) \wedge \textsc{Phone}(\theta_{pd}) \wedge n = \textsc{no}(\theta_{pd})) \\ & \quad \wedge (\textbf{Know}(\textsc{ac}(\textbf{here}) = \textsc{ac}(\textsc{pos}(\theta_{pd}))) \vee \\ & \qquad \exists c\, \textbf{Know}(\textsc{ac}(\textbf{here}) \neq \textsc{ac}(\textsc{pos}(\theta_{pd})) \wedge c = \textsc{ac}(\textsc{pos}(\theta_{pd})))) \supset \\ & \quad \textbf{Can}(\textsc{call}(\theta_{pd}), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d)) \end{aligned}$$

Proof:

$$\begin{aligned} & \textbf{Can}(\textsc{call}(\theta_{pd}), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d)) \\ & \overset{\text{def}}{=} \textbf{Can}(\textbf{if}(\textsc{ac}(\textbf{here}) = \textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{dialloc}(\textsc{no}(\theta_{pd})), \textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{no}(\theta_{pd}))), \\ & \qquad \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d)) \\ & \qquad\qquad\qquad\qquad\qquad\qquad \text{(by the definition of CALL)} \\ & \overset{\text{def}}{=} (\textbf{Know}(\textsc{ac}(\textbf{here}) = \textsc{ac}(\textsc{pos}(\theta_{pd}))) \wedge \\ & \quad \textbf{Can}(\textsc{dialloc}(\textsc{no}(\theta_{pd})), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d))) \vee \\ & \quad (\textbf{Know}(\textsc{ac}(\textbf{here}) \neq \textsc{ac}(\textsc{pos}(\theta_{pd}))) \wedge \\ & \quad \textbf{Can}(\textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{no}(\theta_{pd})), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d))) \\ & \qquad\qquad\qquad\qquad\qquad\qquad \text{(by the definition of } \textbf{Can}) \end{aligned}$$

By propositions 5.1.1 and 5.1.2, $\models \textbf{Know}(\varphi_1 \wedge \varphi_2) \equiv \textbf{Know}(\varphi_1) \wedge \textbf{Know}(\varphi_2)$ and propositional reasoning ($\models (\varphi_1 \supset \varphi_2) \wedge (\varphi_3 \supset \varphi_4) \supset (\varphi_1 \vee \varphi_3 \supset \varphi_2 \vee \varphi_4)$), it must be the case that:

$$\begin{aligned} & \models \exists n\, \textbf{Know}(\exists p_s(\textsc{Phone}(p_s) \wedge \textsc{pos}(p_s) = \textbf{here} \wedge \neg\textsc{Inuse}(p_s)) \wedge \\ & \qquad \textsc{Phone}(\theta_{pd}) \wedge n = \textsc{no}(\theta_{pd}) \wedge \textsc{ac}(\textbf{here}) = \textsc{ac}(\textsc{pos}(\theta_{pd}))) \vee \\ & \quad \exists n \exists c \textbf{Know}(\exists p_s(\textsc{Phone}(p_s) \wedge \textsc{pos}(p_s) = \textbf{here} \wedge \neg\textsc{Inuse}(p_s)) \wedge \\ & \qquad \textsc{Phone}(\theta_{pd}) \wedge n = \textsc{no}(\theta_{pd}) \wedge c = \textsc{ac}(\textsc{pos}(\theta_{pd})) \wedge \textsc{ac}(\textbf{here}) \neq \textsc{ac}(\textsc{pos}(\theta_{pd}))) \\ & \quad \supset (\textbf{Know}(\textsc{ac}(\textbf{here}) = \textsc{ac}(\textsc{pos}(\theta_{pd}))) \wedge \\ & \qquad \textbf{Can}(\textsc{dialloc}(\textsc{no}(\theta_{pd})), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d))) \vee \\ & \quad (\textbf{Know}(\textsc{ac}(\textbf{here}) \neq \textsc{ac}(\textsc{pos}(\theta_{pd}))) \wedge \\ & \qquad \textbf{Can}(\textsc{dialld}(\textsc{ac}(\textsc{pos}(\theta_{pd})), \textsc{no}(\theta_{pd})), \exists p_s \exists p_d\, \textsc{Connected}(p_s, p_d))) \end{aligned}$$

By $\models \mathbf{Know}(\varphi_1 \wedge \varphi_2) \equiv \mathbf{Know}(\varphi_1) \wedge \mathbf{Know}(\varphi_2)$ and elementary reasoning about quantifiers, we must also have that

$$\begin{aligned}
\models&(\exists n \mathbf{Know}(\exists p_s(\mathrm{PHONE}(p_s) \wedge \mathrm{POS}(p_s) = \mathbf{here} \wedge \neg\mathrm{INUSE}(p_s)) \wedge \mathrm{PHONE}(\theta_{pd}) \wedge n = \mathrm{NO}(\theta_{pd}))\\
&\quad \wedge \mathbf{Know}(\mathrm{AC}(\mathbf{here}) = \mathrm{AC}(\mathrm{POS}(\theta_{pd})))) \vee\\
&(\exists n \mathbf{Know}(\exists p_s(\mathrm{PHONE}(p_s) \wedge \mathrm{POS}(p_s) = \mathbf{here} \wedge \neg\mathrm{INUSE}(p_s)) \wedge \mathrm{PHONE}(\theta_{pd}) \wedge n = \mathrm{NO}(\theta_{pd}))\\
&\quad \wedge \exists c\, \mathbf{Know}(\mathrm{AC}(\mathbf{here}) \neq \mathrm{AC}(\mathrm{POS}(\theta_{pd})) \wedge c = \mathrm{AC}(\mathrm{POS}(\theta_{pd})))) \supset\\
&(\mathbf{Know}(\mathrm{AC}(\mathbf{here}) = \mathrm{AC}(\mathrm{POS}(\theta_{pd}))) \wedge\\
&\ \mathbf{Can}(\mathrm{DIALLOC}(\mathrm{NO}(\theta_{pd})), \exists p_s \exists p_d\, \mathrm{CONNECTED}(p_s, p_d))) \vee\\
&(\mathbf{Know}(\mathrm{AC}(\mathbf{here}) \neq \mathrm{AC}(\mathrm{POS}(\theta_{pd}))) \wedge\\
&\ \mathbf{Can}(\mathrm{DIALLD}(\mathrm{AC}(\mathrm{POS}(\theta_{pd})), \mathrm{NO}(\theta_{pd})), \exists p_s \exists p_d\, \mathrm{CONNECTED}(p_s, p_d)))
\end{aligned}$$

Then since $\models (\varphi_1 \wedge \varphi_2) \vee (\varphi_1 \wedge \varphi_3) \equiv \varphi_1 \wedge (\varphi_2 \vee \varphi_3)$, it must also be the case that

$$\begin{aligned}
\models&\exists n \mathbf{Know}(\exists p_s(\mathrm{PHONE}(p_s) \wedge \mathrm{POS}(p_s) = \mathbf{here} \wedge \neg\mathrm{INUSE}(p_s)) \wedge \mathrm{PHONE}(\theta_{pd}) \wedge n = \mathrm{NO}(\theta_{pd}))\\
&\quad \wedge (\mathbf{Know}(\mathrm{AC}(\mathbf{here}) = \mathrm{AC}(\mathrm{POS}(\theta_{pd}))) \vee\\
&\qquad \exists c\, \mathbf{Know}(\mathrm{AC}(\mathbf{here}) \neq \mathrm{AC}(\mathrm{POS}(\theta_{pd})) \wedge c = \mathrm{AC}(\mathrm{POS}(\theta_{pd})))) \supset\\
&(\mathbf{Know}(\mathrm{AC}(\mathbf{here}) = \mathrm{AC}(\mathrm{POS}(\theta_{pd}))) \wedge\\
&\ \mathbf{Can}(\mathrm{DIALLOC}(\mathrm{NO}(\theta_{pd})), \exists p_s \exists p_d\, \mathrm{CONNECTED}(p_s, p_d))) \vee\\
&(\mathbf{Know}(\mathrm{AC}(\mathbf{here}) \neq \mathrm{AC}(\mathrm{POS}(\theta_{pd}))) \wedge\\
&\ \mathbf{Can}(\mathrm{DIALLD}(\mathrm{AC}(\mathrm{POS}(\theta_{pd})), \mathrm{NO}(\theta_{pd})), \exists p_s \exists p_d\, \mathrm{CONNECTED}(p_s, p_d)))
\end{aligned}$$

■

This result shows that our earlier description of the knowledge prerequisites of the "making a phone call" action as formalized here was accurate. It is clear that this characterization of the knowledge prerequisites of the action is much more reasonable than that yielded by the obvious approach described at the outset.

5.2 Traversing a Data Structure

We mentioned earlier that the notion of indexical knowledge is quite abstract; it need not involve relativity to a point of view into any kind of *physical* space. The "space" in which the agent is located, in which he moves about, could be an abstraction, a purely conceptual structure. The computational domain contains a wealth of instances that are naturally viewed as involving some kind of agent navigating in an abstract space, say a data structure or a search space in problem solving.

Consider the following example: an agent is using a stack to accomplish some task, say reversing a list. There is strong analogy between some of the robotics situations discussed in chapter 4 and this situation (especially if the stack is stored as a contiguous array). According to the stack access discipline, only the top element of the stack is "visible"; analogously, the robot of chapter 4 could sense only what was **here**. When one pops the stack, what happens is similar to the robot clearing the current square and going forward onto the next square. When one pushes an new data element on the stack, this is similar to our robot going backward one square and putting (or writing) the element down on that square. It would be straightforward to formalize this indexical interpretation of the stack situation: we would take **here** to stand for a position in the stack area; the immediate accessibility of the top of the stack would be captured by the agent's knowing what element is **here**; the push and pop operations would change the position that **here** stands for, their effects being specified in relative terms.

We could develop the stack example in details, but we will rather look at another example, one involving a "heap", as one sees in the heapsort algorithm (Aho, Hopcroft and Ullman 1974). A heap is a binary tree that satisfies the "heap property", that is, where the key stored at any node is greater or equal to the ones stored at its left and right sons (if they exist). By transitivity, this means that the key at a node must be at least as large as any other key in the subtree rooted there. In this case too, it is very natural to describe the algorithm in terms of an "agent" being at some node and navigating down the tree. This example is probably more interesting than the stack one because the "space" involved has a very different topology from that of physical space; one is not moving along a line, or on a plane, or in 3-dimensional space, but along the edges of a tree.

In the heapsort algorithm, the heap is stored in an array where the left and right sons of node $A(i)$ are in $A(2i)$ and $A(2i+1)$ respectively. The algorithm is built around a subprocedure, call it HEAPIFY, that takes a binary tree where the proper subtrees of the root (the ones that start with its sons) satisfy the heap property, but where the whole tree might not, and sifts the element at the root down until the whole tree is a heap (this takes $O(\log(n))$ time where n is the number of nodes). The main procedure of heapsort works in two phases: first, one starts at the end of the array and keeps enlarging the trees by adding elements from the beginning (new roots) and running HEAPIFY on them until the whole array is a heap; then one keeps exchanging the root (the largest element) with the element at the end of the heap part of the array, shrinking the tree by one, and running HEAPIFY on the result, until the whole array is sorted.

We will develop a formalization of the HEAPIFY part of the algorithm; this should be sufficient to demonstrate the applicability of our framework and the notion of indexical knowledge to this kind of situation. Our specification of the HEAPIFY procedure goes as follows:

Definition 5.2.1

$$\begin{aligned}\textsc{heapify}_k \overset{\text{def}}{=} \mathbf{while}_k(&\textsc{SmallerThanSon},\\ &\mathbf{if}(\textsc{LeftSonLargest},\\ &\quad(\textsc{swapKeys}(\textsc{left}(\mathbf{here}), \mathbf{here}); \textsc{goLeft}),\\ &\quad(\textsc{swapKeys}(\textsc{right}(\mathbf{here}), \mathbf{here}); \textsc{goRight})))\end{aligned}$$

$$\begin{aligned}\textsc{SmallerThanSon} \overset{\text{def}}{=} (&\textsc{IsLeft}(\mathbf{here}) \wedge \textsc{key}(\textsc{left}(\mathbf{here})) > \textsc{key}(\mathbf{here})) \vee\\ &\textsc{IsRight}(\mathbf{here}) \wedge \textsc{key}(\textsc{right}(\mathbf{here})) > \textsc{key}(\mathbf{here}))\end{aligned}$$

$$\textsc{LeftSonLargest} \overset{\text{def}}{=} \textsc{key}(\textsc{left}(\mathbf{here})) > \textsc{key}(\textsc{right}(\mathbf{here}))$$

Doing HEAPIFY involves repeatedly swapping the key of the node where one is (i.e., the node at position **here**) with that of its largest son and then moving down to this son; one keeps doing this for as long as the key at node **here** is smaller than that of either of its sons (but at most k times).

We would like to prove something like the following conjecture; it says that if it is known that the tree rooted **here** has a depth of at most k, that both the left and right subtrees are heaps and if various facts about the current node and its immediate sons are known (KNOWLOCALSIT), then one is able to achieve the goal of making the tree rooted **here** into a heap by doing $\textsc{heapify}_k$:

Conjecture 5.2.1

For all $k \in \mathbb{N}$,

$$\begin{aligned}&\models \mathbf{Know}(\textsc{depth}(\mathbf{here}) \leq k \wedge \textsc{Heap}(\textsc{left}(\mathbf{here})) \wedge \textsc{Heap}(\textsc{right}(\mathbf{here}))\\ &\quad \wedge \textsc{KnowLocalSit} \supset\\ &\quad \mathbf{Can}(\textsc{heapify}_k, \mathbf{Know}(\exists p(\mathbf{DoneWhen}(\textsc{heapify}_k, p = \mathbf{here}) \wedge \textsc{Heap}(p))))\end{aligned}$$

Definition 5.2.2

$$\begin{aligned}\textsc{KnowLocalSit} \overset{\text{def}}{=}\ &\exists k\mathbf{Know}(k = \textsc{key}(\mathbf{here})) \wedge\\ &\mathbf{Kwhether}(\textsc{IsLeft}(\mathbf{here})) \wedge \mathbf{Kwhether}(\textsc{IsRight}(\mathbf{here})) \wedge\\ &(\textsc{IsLeft}(\mathbf{here}) \supset \exists k\mathbf{Know}(k = \textsc{key}(\textsc{left}(\mathbf{here})))) \wedge\\ &(\textsc{IsRight}(\mathbf{here}) \supset \exists k\mathbf{Know}(k = \textsc{key}(\textsc{right}(\mathbf{here}))))\end{aligned}$$

KNOWLOCALSIT holds if and only if the agent knows what the key at its current position is, knows whether the current node has a left and/or right son, and if it does, knows what their keys are.

We will not develop a full formalization of this situation, but we will describe how it would go and sketch how one would show that conjecture 5.2.1 holds.[2]

The effects of GOLEFT and GORIGHT are formalized as follows:

Assumption 5.2.1 (Effects of GOLEFT)

$$\models \mathbf{Res}(\textsc{goLeft}, \exists p(\mathbf{DoneWhen}(\textsc{goLeft}, \mathbf{here} = p) \wedge \mathbf{here} = \textsc{left}(p)) \wedge \textsc{KnowLocalSit})$$

Assumption 5.2.2 (Effects of GORIGHT)

$$\models \mathbf{Res}(\textsc{goRight}, \exists p(\mathbf{DoneWhen}(\textsc{goRight}, \mathbf{here} = p) \wedge \mathbf{here} = \textsc{right}(p)) \wedge \textsc{KnowLocalSit})$$

We would also need frame assumptions stating that these actions change no other aspects of the situation. Assumptions specifying the SWAPKEYS action would also be needed.

Now, let us show how a proof of our main conjecture (5.2.1) would go. First, we should be able to show that given what the agent knows about the current node and its sons (KNOWLOCALSIT), he must know whether the tests of the while-loop and the conditional hold:

Conjecture 5.2.2

$$\models \textsc{KnowLocalSit} \supset \mathbf{Kwhether}(\textsc{SmallerThanSon}) \wedge \mathbf{Kwhether}(\textsc{LeftSonLargest})$$

It should also be possible to show that given what the agent knows, if the test of the while-loop is false, then he must know that the tree rooted **here** is already a heap; this is what the following conjecture says:

[2]We feel that the earlier examples have adequately shown how one goes about developing formal proofs in the framework. At this point, it seems more important to show the applicability of the framework in a wide range of situations.

Conjecture 5.2.3

For all $k \in \mathbb{N}$,

$$\models \textbf{Know}(\text{DEPTH}(\textbf{here}) \leq k \wedge \text{HEAP}(\text{LEFT}(\textbf{here})) \wedge \text{HEAP}(\text{RIGHT}(\textbf{here})) \wedge \neg\text{SMALLERTHANSON}) \wedge \text{KNOWLOCALSIT} \supset \textbf{Know}(\text{HEAP}(\textbf{here}))$$

The next two conjectures cover each alternative of the conditional:

Conjecture 5.2.4

For all $k \in \mathbb{N}$,

$$\begin{aligned}\models\ & \textbf{Know}(\text{DEPTH}(\textbf{here}) \leq k \wedge \text{HEAP}(\text{LEFT}(\textbf{here})) \wedge \text{HEAP}(\text{RIGHT}(\textbf{here})) \wedge \\ &\quad \text{SMALLERTHANSON} \wedge \text{LEFTSONLARGEST}) \wedge \text{KNOWLOCALSIT} \supset \\ & \textbf{Can}((\text{SWAPKEYS}(\text{LEFT}(\textbf{here}), \textbf{here}); \text{GOLEFT}), \text{KNOWLOCALSIT} \wedge \\ &\quad \textbf{Know}(\text{DEPTH}(\textbf{here}) \leq k-1 \wedge \text{HEAP}(\text{LEFT}(\textbf{here})) \wedge \text{HEAP}(\text{RIGHT}(\textbf{here})) \wedge \\ &\qquad \exists p(\textbf{DoneWhen}((\text{SWAPKEYS}(\text{LEFT}(\textbf{here}), \textbf{here}); \text{GOLEFT}), p = \textbf{here}) \wedge \\ &\qquad\quad \text{HEAP}(\text{RIGHT}(p)) \wedge \text{KEY}(\textbf{here}) > \text{KEY}(\text{RIGHT}(p)) \wedge \text{KEY}(p) > \text{KEY}(\textbf{here}))))\end{aligned}$$

Conjecture 5.2.5

For all $k \in \mathbb{N}$,

$$\begin{aligned}\models\ & \textbf{Know}(\text{DEPTH}(\textbf{here}) \leq k \wedge \text{HEAP}(\text{LEFT}(\textbf{here})) \wedge \text{HEAP}(\text{RIGHT}(\textbf{here})) \wedge \\ &\quad \text{SMALLERTHANSON} \wedge \neg\text{LEFTSONLARGEST}) \wedge \text{KNOWLOCALSIT} \supset \\ & \textbf{Can}((\text{SWAPKEYS}(\text{RIGHT}(\textbf{here}), \textbf{here}); \text{GORIGHT}), \text{KNOWLOCALSIT} \wedge \\ &\quad \textbf{Know}(\text{DEPTH}(\textbf{here}) \leq k-1 \wedge \text{HEAP}(\text{LEFT}(\textbf{here})) \wedge \text{HEAP}(\text{RIGHT}(\textbf{here})) \wedge \\ &\qquad \exists p(\textbf{DoneWhen}((\text{SWAPKEYS}(\text{RIGHT}(\textbf{here}), \textbf{here}); \text{GORIGHT}), p = \textbf{here}) \\ &\qquad\quad \wedge \text{HEAP}(\text{LEFT}(p)) \wedge \text{KEY}(\textbf{here}) \geq \text{KEY}(\text{LEFT}(p)) \wedge \text{KEY}(p) > \text{KEY}(\textbf{here}))))\end{aligned}$$

Given these results, one should then be able to develop an induction proof of the main conjecture (5.2.1).

5.3 Temporally Constrained Goals

In chapter 4, we used spatial examples to motivate the distinction between indexical knowledge and *de re* knowledge and its usefulness in accounting for ability. We described cases where the agent is required to know his absolute position, other cases where he need only know where something is relative to himself, that is, know its relative position, and other cases where neither kind of knowledge is needed — the agent needs only know a navigation procedure that gets him where he wants. In this section, we will show that the same kind of distinctions are useful in handling goals that require temporal knowledge. In section 5.3.2, we develop an example where one must know what time it is to be able to achieve a goal; it involves an agent who must ensure that a room is locked after 5 p.m. In the following section, we describe a case where knowing how much time has elapsed since some event occurred is sufficient for ability; the example involves an agent who wants to soft-boil an egg. Finally in section 5.3.4, we describe a case where an agent knows that a process has gone on for an appropriate length of time, but not how long that was, simply by monitoring the situation; the example involves an agent who wants to fry a fish filet for just the right amount of time and who achieves this by frying it until its flesh is no longer transparent.

These examples, involving situations that are in no way unusual, constitute strong evidence that the distinction between indexical and objective knowledge is an essential one for a formalization of knowledge and ability, one that cannot be mimicked or defined away. This is discussed further in section 5.5.

5.3.1 Preliminaries

For the examples in this section, we assume that the domain of times is (or is isomorphic to) the integers. As in chapter 4, we will assume that arithmetic constants and function symbols are rigid and that all facts about integer arithmetic that we need are valid (assumptions 4.9, 4.10, and 4.11).[3]

The first two examples require the agent to keep track of time by looking at a clock. We say that a clock is CORRECT (at some point in time) if and only if the time it indicates, its VALUE, is the current time. Formally, this amounts to:

Definition 5.3.1 $\mathrm{CORRECT}(c) \overset{\mathrm{def}}{=} \mathrm{VALUE}(c) = \mathbf{now}$

We use the predicate ACCURATE to say that a clock is working accurately at a given point in time. This attribute is important because if a clock works accurately over a time interval, it correctly measures how much time passes over the interval. The following assumption formalizes this:

[3] Lipson (1981) gives an axiomatization of the integers.

Assumption 5.3.1

$$\models \forall t_1 \forall t_2 \forall c \forall t_c (\mathbf{At}(t_1, \textsc{value}(c) = t_c) \land \forall t_i (t_1 \leq t_i \leq t_2 \supset \mathbf{At}(t_i, \textsc{Accurate}(c))) \\ \supset \mathbf{At}(t_2, \textsc{value}(c) = t_c + t_2 - t_1)$$

The next conjecture expresses a useful consequence of this; it says that if a clock is CORRECT and works accurately from **now** to some future time t, then it will still be CORRECT at time t.

Conjecture 5.3.1

$$\models \forall t(\textsc{Correct}(c) \land \forall t'(\mathbf{now} \leq t' \leq t \supset \mathbf{At}(t', \textsc{Accurate}(c))) \supset \mathbf{At}(t, \textsc{Correct}(c)))$$

5.3.2 Locking Up at 5 p.m.

Our first temporal example concerns a case where an agent is required to know what time it is in order to be able to achieve a goal. It involves an agent who is supposed to control access to a room (shop, safe, ATM access room, etc.); in particular, he is supposed to ensure that the room is locked after 5 p.m. The agent's goal, OBSERVESCHED, is specified formally below; it is stated to hold if and only if it is presently between time 500 and time 550, the door is locked, and it was unlocked at all times before 500. We also state that the agent's plan, WAITLOCKUP, involves first looking at clock c, then for as long as it is not yet time 500, to keep waiting and looking at the clock, and then when time 500 arrives, to lock up (the number of iterations being at most 250).

Definition 5.3.2

$$\textsc{ObserveSched} \overset{\text{def}}{=} 500 \leq \mathbf{now} \leq 550 \land \textsc{Locked} \land \forall t(t < 500 \supset \mathbf{At}(t, \neg\textsc{Locked}))$$

$$\textsc{waitLockUp}(\theta^i_c) \overset{\text{def}}{=} \textsc{lookAt}(\theta^i_c); \mathbf{while}_{250}(\mathbf{now} \leq 500, (\textsc{wait}; \textsc{lookAt}(\theta^i_c))); \textsc{lockUp}$$

We would like to show that if the agent knows that clock c is correct and accurate, that the room has always been unlocked, and that it is presently between time 0 and time 500, then he is able to achieve the goal OBSERVESCHED by doing the action WAITLOCKUP(c). This is stated formally in the following conjecture:

Conjecture 5.3.2

$$\models \forall c(\mathbf{Know}(\textsc{Correct}(c) \land \textsc{Accurate}(c) \land \mathbf{AllPastN}(\neg\textsc{Locked}) \land 0 \leq \mathbf{now} < 500) \supset \mathbf{Can}(\textsc{waitLockUp}(c), \textsc{ObserveSched}))$$

We will give a sketch of how this could be proven, but first, we must specify the relevant aspects of the domain, in particular, the properties of the actions involved. First note that we must specify an appropriate limit on the duration of the WAIT, LOOKAT, and LOCKUP actions, otherwise they might take so much time as to prevent the goal from being achieved. But note also that we cannot have these actions always take a fixed length of time that is known to the agent, for otherwise, he could use one of them to measure time and thus dispense with the clock. We will thus specify that these basic actions take a indeterminate amount of time that must be between 1 and 10 units. The following assumption states this for action WAIT in a slightly indirect way; it says that WAIT results in the fact that between 1 and 10 units have passed since the action was undertaken:

Assumption 5.3.2 (Effects of WAIT)

$$\models \mathbf{Res}(\textsc{wait}, \exists t(\mathbf{now} - 10 \leq t \leq \mathbf{now} - 1 \land \mathbf{DoneFrom}(\textsc{wait}, t)))$$

Note that this implicitly says that it is always physically possible to do WAIT (by the definition of **Res**). Also, since we assume that this is valid, it must be common knowledge that these limits on the duration of WAIT apply. We assume that all the basic actions introduced in this section are "solid", that is, that any overlapping instances of them must be the same instance (in the sense that they have the same endpoints); this specified as follows:

Assumption 5.1 (Basic actions are solid)

$$\models \forall t_{s1} \forall t_{e1} \forall t_{s2} \forall t_{e2}(\mathbf{DoneFromTo}(\theta^d, t_{s1}, t_{e1}) \land \mathbf{DoneFromTo}(\theta^d, t_{s2}, t_{e2}) \land \textsc{Overlaps}(t_{s1}, t_{e1}, t_{s2}, t_{e2}) \supset t_{s1} = t_{s2} \land t_{e1} = t_{e2})$$

The latter assumption is necessary if, for instance, assumption 5.3.2 is to mean that *all* instances of WAIT must take between 1 and 10 units.

We also assume that agents always know what the WAIT action is:

Assumption 5.3.3 (WAIT is known) $\models \exists d\mathbf{Know}(d = \text{WAIT})$

We also have the following frame assumption, which says that WAIT does not affect whether the room is locked or not:

Assumption 5.3.4 (WAIT does not affect LOCKED)

$$\models \mathbf{AfterNec}(\text{WAIT}, \exists t(\mathbf{DoneFrom}(\text{WAIT}, t) \wedge \\ \forall t'(t < t' \leq \mathbf{now} \supset (\mathbf{At}(t', \text{LOCKED}) \equiv \mathbf{At}(t, \text{LOCKED}))))$$

As a consequence of this, it should be the case that if the room has always been unlocked when the agent starts doing WAIT, then it must always have been unlocked afterwards:

Conjecture 5.3.3

$$\models \mathbf{AfterNec}(\text{WAIT}, \mathbf{DoneWhen}(\text{WAIT}, \mathbf{AllPastN}(\neg\text{LOCKED})) \supset \mathbf{AllPastN}(\neg\text{LOCKED}))$$

Similarly, we also assume that WAIT does not affect whether or not a clock is working accurately:

Assumption 5.3.5 (WAIT does not affect ACCURATE)

$$\models \mathbf{AfterNec}(\text{WAIT}, \exists t(\mathbf{DoneFrom}(\text{WAIT}, t) \wedge \\ \forall t'(t < t' \leq \mathbf{now} \supset \forall c(\mathbf{At}(t', \text{ACCURATE}(c)) \equiv \mathbf{At}(t, \text{ACCURATE}(c)))))$$

Now, let's specify the properties of the action of "looking at a clock". We assume that LOOKAT also takes between 1 and 10 time units. We also assume that after doing LOOKAT, the agent knows what time the clock is indicating as the action ends.[4] This yields the following:

Assumption 5.3.6 (Effects of LOOKAT)

$$\models \forall c\mathbf{Res}(\text{LOOKAT}(c), \exists t(\mathbf{now} - 10 \leq t \leq \mathbf{now} - 1 \wedge \mathbf{DoneFrom}(\text{WAIT}, t)) \\ \wedge \exists t\mathbf{Know}(t = \text{VALUE}(c)))$$

As a consequence of this assumption, it should be the case that if the agent looks at a clock and knows that this clock is correct, then he must know what time it is:

[4]It would be more realistic to say that the agent gets to know what time the clock indicated at some point in the performance of the action; but this would needlessly complicate the example.

Conjecture 5.3.4

$$\models \forall c \mathbf{AfterNec}(\textsc{lookAt}(c), \mathbf{Know}(\textsc{Correct}(c)) \supset \exists t \mathbf{Know}(t = \mathbf{now}))$$

We assume that agents always know what the action of looking at a clock is when they know which clock is involved:

Assumption 5.3.7 (LOOKAT is known) $\models \forall c \exists d \mathbf{Know}(d = \textsc{lookAt}(c))$

This is not realistic; it would be better to require the agent to know the position of the clock relative to himself, as we did for the source phone in the example of section 5.1; but this is not central to the formalization of this example. As for WAIT, we assume that LOOKAT does not affect whether the room is locked or not:

Assumption 5.3.8 (LOOKAT does not affect LOCKED)

$$\models \forall c \mathbf{AfterNec}(\textsc{lookAt}(c), \exists t(\mathbf{DoneFrom}(\textsc{lookAt}(c), t) \wedge \\ \forall t'(t < t' \leq \mathbf{now} \supset (\mathbf{At}(t', \textsc{Locked}) \equiv \mathbf{At}(t, \textsc{Locked}))))$$

Thus it should be the case that if the room has always been unlocked when LOOKAT is done, then it must always have been unlocked afterwards.

Conjecture 5.3.5

$$\models \forall c \mathbf{AfterNec}(\textsc{lookAt}(c), \\ \mathbf{DoneWhen}(\textsc{lookAt}(c), \mathbf{AllPastN}(\neg\textsc{Locked})) \supset \mathbf{AllPastN}(\neg\textsc{Locked}))$$

We also assume that LOOKAT does not affect whether or not any clock is working accurately:

Assumption 5.3.9 (LOOKAT does not affect ACCURATE)

$$\models \forall c \mathbf{AfterNec}(\textsc{lookAt}(c), \exists t(\mathbf{DoneFrom}(\textsc{wait}, t) \wedge \\ \forall t'(t < t' \leq \mathbf{now} \supset \forall c'(\mathbf{At}(t', \textsc{Accurate}(c')) \equiv \mathbf{At}(t, \textsc{Accurate}(c'))))))$$

Finally, we assume that the action LOCKUP also takes between 1 and 10 time units and that it results in the room being locked:

Assumption 5.3.10 (Effects of LOCKUP)

$$\models \mathbf{Res}(\textsc{lockUp}, \exists t(\mathbf{now} - 10 \leq t \leq \mathbf{now} - 1 \wedge \mathbf{DoneFrom}(\textsc{lockUp}, t)) \wedge \textsc{Locked})$$

And we assume that agents always know how to do $\textsc{lockUp}$:

Assumption 5.3.11 ($\textsc{lockUp}$ is known) $\models \exists d\mathbf{Know}(d = \textsc{lockUp})$

We are now ready to sketch the proof of our main result, conjecture 5.3.2. First, let $\mathrm{CAAU}(\theta^i_c)$ stand for the claim that clock θ^i_c is correct and accurate and that the room has always been unlocked:

Definition 5.3.3

$$\mathrm{CAAU}(\theta^i_c) \overset{\text{def}}{=} \textsc{Correct}(\theta^i_c) \wedge \textsc{Accurate}(\theta^i_c) \wedge \mathbf{AllPastN}(\neg\textsc{Locked})$$

Now let us consider the $\textsc{lookAt}$ action. From the assumptions, it should be possible to prove the following conjecture, which says that if the agent knows it is presently between time t_l and time t_h, that clock c is correct and accurate, and that the room has always been unlocked, then by looking at c, he is able to achieve the goal of knowing what time it is and knowing that it is presently between time $t_l + 1$ and time $t_h + 10$, that clock c is correct and accurate, and that the room has always been unlocked.

Conjecture 5.3.6

$$\models \forall t_l \forall t_h \forall c(\mathbf{Know}(t_l \leq \mathbf{now} \leq t_h \wedge \mathrm{CAAU}(c)) \supset$$
$$\mathbf{Can}(\textsc{lookAt}(c), \exists t\mathbf{Know}(t = \mathbf{now} \wedge t_l + 1 \leq \mathbf{now} \leq t_h + 10 \wedge \mathrm{CAAU}(c))))$$

By instantiating this with the time bounds that apply initially, we should get that by looking at c, the agent is initially able to achieve the goal of knowing what time it is and knowing that it is presently between time 1 and time 510, that clock c is correct and accurate, and that the room has always been unlocked.

Conjecture 5.3.7

$$\models \forall c(\mathbf{Know}(0 \leq \mathbf{now} \leq 500 \wedge \mathrm{CAAU}(c)) \supset$$
$$\mathbf{Can}(\textsc{lookAt}(c), \exists t\mathbf{Know}(t = \mathbf{now} \wedge 1 \leq \mathbf{now} \leq 510 \wedge \mathrm{CAAU}(c))))$$

Next, we need to deal with the while-loop action. First, we should be able to prove the following conjecture about the $\textsc{wait}$ action; it says that if the agent knows it is presently between time t_l and time t_h, that clock c is correct and accurate, and that the room has always been unlocked, then by doing $\textsc{wait}$, he is able to achieve the goal of knowing that it is presently between time $t_l + 1$ and time $t_h + 10$, that clock c is correct and accurate, and that the room has always been unlocked:

Conjecture 5.3.8

$$\models \forall t_l \forall t_h \forall c(\mathbf{Know}(t_l \leq \mathbf{now} \leq t_h \wedge \mathrm{CAAU}(c)) \supset$$
$$\mathbf{Can}(\textsc{wait}, \mathbf{Know}(t_l + 1 \leq \mathbf{now} \leq t_h + 10 \wedge \mathrm{CAAU}(c))))$$

By chaining this result with conjecture 5.3.6, we should obtain that if the agent knows it is presently between time t_l and time t_h, that clock c is correct and accurate, and that the room has always been unlocked, then by doing WAIT and then looking at c, he is able to achieve the goal of knowing what time it is and knowing that it is presently between time $t_l + 2$ and time $t_h + 20$, that clock c is correct and accurate, and that the room has always been unlocked:

Conjecture 5.3.9

$$\models \forall t_l \forall t_h \forall c(\mathbf{Know}(t_l \leq \mathbf{now} \leq t_h \wedge \mathrm{CAAU}(c)) \supset$$
$$\mathbf{Can}((\textsc{wait}; \textsc{lookAt}(c)),$$
$$\exists t \mathbf{Know}(t = \mathbf{now} \wedge t_l + 2 \leq \mathbf{now} \leq t_h + 20 \wedge \mathrm{CAAU}(c))))$$

Then by induction on k, we should be able show that if the agent knows what time it is and knows that it is presently between time t_l and time 520, that clock c is correct and accurate, and that the room has always been unlocked, then by repeatedly waiting and looking at c for as long as it is not yet time 500 (up to k times), he is able to achieve the goal of knowing what time it is and knowing that the current time is greater or equal to the least of time $2\mathsf{k} + t_l$ and time 500, and less or equal to time 520, and that clock c is correct and accurate, and that the room has always been unlocked:

Conjecture 5.3.10

For all $\mathsf{k} \in \mathbb{N}$,
$$\models \forall t_l \forall c(\exists t \mathbf{Know}(t = \mathbf{now} \wedge t_l \leq \mathbf{now} \leq 520 \wedge \mathrm{CAAU}(c)) \supset$$
$$\mathbf{Can}(\mathbf{while}_{\mathsf{k}}(\mathbf{now} \leq 500, (\textsc{wait}; \textsc{lookAt}(c))),$$
$$\exists t \mathbf{Know}(t = \mathbf{now} \wedge \textsc{min}(2\mathsf{k} + t_l, 500) \leq \mathbf{now} \leq 520 \wedge \mathrm{CAAU}(c))))$$

To sketch the proof of this: in the base case ($\mathsf{k} = 0$), the action amounts to **skip**, in which case we must show that the agent already knows that the goal holds, and this follows from the antecedent by proposition 3.14; for the induction case, the antecedent clearly implies that the agent knows whether the loop condition holds, and then in case it does not hold, then the action again amounts to **skip**, and this and the antecedent imply that the goal is known to hold, and in case loop condition does hold, we can prove the result from conjecture 5.3.9 and the induction hypothesis.

We can then instantiate the above conjecture to get the following, which says that if the agent knows what time it is and knows that it is presently between time 1 and time 520, that clock c is correct and accurate, and that the room has always been unlocked, then by repeatedly waiting and looking at c for as long as it is not yet time 500 (up to 250 times), he is able to achieve the goal of knowing what time it is and knowing that it is presently between time 500 and time 520, that clock c is correct and accurate, and that the room has always been unlocked:

Conjecture 5.3.11

$$\begin{aligned}&\models \forall c(\exists t\mathbf{Know}(t = \mathbf{now} \land 1 \leq \mathbf{now} \leq 520 \land \mathrm{CAAU}(c)) \supset \\ &\qquad \mathbf{Can}(\mathbf{while}_{250}(\mathbf{now} \leq 500, (\textsc{wait}; \textsc{lookAt}(c))), \\ &\qquad\qquad \exists t\mathbf{Know}(t = \mathbf{now} \land 500 \leq \mathbf{now} \leq 520 \land \mathrm{CAAU}(c))))\end{aligned}$$

Next we should be able to prove the following conjecture about action LOCKUP; it says that if the agent knows that it is presently between time 500 and time 520, that clock c is correct and accurate, and that the room has always been unlocked, then by locking up, the agent is able to achieve the main goal of observing his schedule:

Conjecture 5.3.12

$$\models \forall c(\mathbf{Know}(500 \leq \mathbf{now} \leq 520 \land \mathrm{CAAU}(c)) \supset \mathbf{Can}(\textsc{lockUp}, \textsc{ObserveSched}))$$

Then by chaining conjectures 5.3.7, 5.3.11, and 5.3.12, it should be easy to prove the main conjecture, which says that if the agent knows that clock c is correct and accurate, that the room has always been locked, and that it is presently between time 0 and time 500, then he is able to achieve the goal OBSERVESCHED by doing the action WAITLOCKUP(c):

Conjecture 5.3.2

$$\begin{aligned}&\models \forall c(\mathbf{Know}(\textsc{Correct}(c) \land \textsc{Accurate}(c) \land \mathbf{AllPastN}(\neg\textsc{Locked}) \land 0 \leq \mathbf{now} < 500) \supset \\ &\qquad \mathbf{Can}(\textsc{waitLockUp}(c), \textsc{ObserveSched}))\end{aligned}$$

5.3.3 Soft-Boiling an Egg

Now let's look at a different example, one that does not require knowing what time it is, one for which knowing how much time has passed since a previous event is sufficient. This makes it analogous to cases where one has to know what the relative position of something is. The example is drawn from the cooking domain. The goal is to soft-boil an egg, that is, to boil it so that it is cooked but that its yolk is still soft. This can normally be done for a large egg by boiling it for approximately five minutes. To be able to achieve this, an agent needs only a clock (or timer) that works accurately; it need not be correct (i.e., on time).

This is formalized in the following conjecture; it says that if the agent knows of a clock c that it is accurate and of an egg e that it is uncooked, then he is able to achieve the goal of having soft-boiled e by doing the action of first putting e to boil, then looking at c, then waiting for about 500 time units, and then removing e from boiling:

Conjecture 5.3.13

$$\models \forall e \forall c(\mathbf{Know}(\textsc{Accurate}(c) \land \textsc{UnCooked}(e)) \supset$$
$$\mathbf{Can}((\textsc{startBoil}(e); \textsc{lookAt}(c); \textsc{wait500u}_{250,0}; \textsc{stopBoil}(e)), \textsc{SoftBoiled}(e)))$$

The goal is defined precisely below. We first say that an egg θ_e^i has just been soft-boiled if and only if it is not currently boiling, it was uncooked at some point in time between 500 and 520 units ago, and for the whole period between then and **now**, it was boiling. Then, we say that an egg θ_e^i is soft-boiled if and only if it was just soft-boiled at some time t_p no later than **now** and it has not been boiling since.

Definition 5.3.4

$$\textsc{JustSoftBoiled}(\theta_e^i) \overset{\text{def}}{=} \neg\textsc{Boiling}(\theta_e^i) \land$$
$$\exists n(500 \leq n \leq 520 \land \mathbf{At}(\mathbf{now} - n, \textsc{UnCooked}(\theta_e^i)) \land$$
$$\forall t(\mathbf{now} - n < t < \mathbf{now} \supset \mathbf{At}(t, \textsc{Boiling}(\theta_e^i))))$$

$$\textsc{SoftBoiled}(\theta_e^i) \overset{\text{def}}{=} \exists t_p(t_p \leq \mathbf{now} \land \mathbf{At}(t_p, \textsc{JustSoftBoiled}(\theta_e^i)) \land$$
$$\forall t(t_p \leq t \leq \mathbf{now} \supset \mathbf{At}(t, \neg\textsc{Boiling}(\theta_e^i))))$$

The action $\textsc{wait500u}_{\mathsf{k,n}}$ is defined below. The definition is similar to that of an ordinary while-loop. Essentially one repeatedly waits and looks at the clock until it indicates 500 units *more than at the beginning.* We cannot specify this action using the standard while form because the termination condition refers back to the time at which the loop was entered. To do this, we must keep track of the number of iterations done, which is the role of parameter n in the definition below. k is a bound on the number of iterations as for ordinary while-loops.

Definition 5.3.5

$$\textsc{wait500u}_{\mathsf{k},\mathsf{n}}(\theta^i_c) \overset{\text{def}}{=} \begin{cases} \textbf{skip} & \text{if } \mathsf{k}=0 \\ \textbf{ifThen}(500\textsc{u}\mathrm{E}_{\mathsf{n}}(c), & \\ \qquad \textsc{wait}; \textsc{lookAt}(\theta^i_c); \textsc{wait500u}_{\mathsf{k}-1,\mathsf{n}+1}(\theta^i_c)) & \text{if } \mathsf{k}>0 \end{cases}$$

$$500\textsc{u}\mathrm{E}_{\mathsf{n}}(\theta^i_c) \overset{\text{def}}{=}$$
$$\exists t(\textbf{DoneWhen}((\textsc{wait}; \textsc{lookAt}(\theta^i_c))^{\mathsf{n}}, \textsc{value}(\theta^i_c) = t) \land \textsc{value}(\theta^i_c) \leq t + 500)$$

Let us then formalize the actions involved in this example and sketch how a proof of conjecture 5.3.13 would go. We begin with action $\textsc{startBoil}$. The next assumption specifies the effects of this action; it says that if some egg e is initially uncooked, then starting to boil e would result in a state of affairs where e is boiling, between 1 and 10 time units have elapsed since the action started, and e was uncooked from the time the action started until the moment before it ended:

Assumption 5.3.12 (Effects of $\textsc{startBoil}$)

$$\begin{aligned} &\models \forall e(\textsc{UnCooked}(e) \supset \\ &\quad \textbf{Res}(\textsc{startBoil}(e), \\ &\qquad \textsc{Boiling}(e) \land \exists t(\textbf{now} - 10 \leq t \leq \textbf{now} - 1 \land \textbf{DoneFrom}(\textsc{startBoil}(e), t) \\ &\qquad\qquad \land \forall t'(t < t' \leq \textbf{now} - 1 \supset \textbf{At}(t', \textsc{UnCooked}(e)))))) \end{aligned}$$

We also assume that agents know how to start boiling some entity e if they know which entity e is:

Assumption 5.3.13 ($\textsc{startBoil}$ is known) $\models \forall e \exists d \textbf{Know}(d = \textsc{startBoil}(e))$

And we also assume that $\textsc{startBoil}$ does not affect the accuracy of any clocks:

Assumption 5.3.14 ($\textsc{startBoil}$ does not affect $\textsc{Accurate}$)

$$\models \forall e \mathbf{AfterNec}(\textsc{startBoil}(e), \exists t(\mathbf{DoneFrom}(\textsc{wait}, t) \land \\ \forall t'(t < t' \leq \mathbf{now} \supset \forall c(\mathbf{At}(t', \textsc{Accurate}(c)) \equiv \mathbf{At}(t, \textsc{Accurate}(c))))))$$

Now let us look at action $\textsc{stopBoil}$. The following assumption specifies the effects of this action; it says that if some egg e is initially boiling, then stopping e from boiling would result in a state of affairs where e is no longer boiling, between 1 and 10 time units have elapsed since the action started, and e was boiling from the time the action started until the moment before it ended:

Assumption 5.3.15 (Effects of $\textsc{stopBoil}$)

$$\models \forall e(\textsc{Boiling}(e) \supset \\ \mathbf{Res}(\textsc{stopBoil}(e), \\ \neg\textsc{Boiling}(e) \land \exists t(\mathbf{now} - 10 \leq t \leq \mathbf{now} - 1 \land \mathbf{DoneFrom}(\textsc{stopBoil}(e), t) \\ \land \forall t'(t < t' \leq \mathbf{now} - 1 \supset \mathbf{At}(t', \textsc{Boiling}(e))))))$$

We also assume that one knows how to stop an entity e from boiling if one knows which entity e is:

Assumption 5.3.16 ($\textsc{stopBoil}$ is known) $\models \forall e \exists d \mathbf{Know}(d = \textsc{stopBoil}(e))$

For the actions $\textsc{wait}$ and $\textsc{lookAt}$, we need to supplement the assumptions stated in the previous section with the following frame assumptions, which say that neither action affects whether any entity is boiling:

Assumption 5.3.17 ($\textsc{wait}$ does not affect $\textsc{Boiling}$)

$$\models \mathbf{AfterNec}(\textsc{wait}, \exists t(\mathbf{DoneFrom}(\textsc{wait}, t) \land \\ \forall t'(t < t' \leq \mathbf{now} \supset \forall e(\mathbf{At}(t', \textsc{Boiling}(e)) \equiv \mathbf{At}(t, \textsc{Boiling}(e))))))$$

Assumption 5.3.18 ($\textsc{lookAt}$ does not affect $\textsc{Boiling}$)

$$\models \forall c \mathbf{AfterNec}(\textsc{LookAt}(c), \exists t(\mathbf{DoneFrom}(\textsc{LookAt}(c), t) \land \\ \forall t'(t < t' \leq \mathbf{now} \supset \forall e(\mathbf{At}(t', \textsc{Boiling}(e)) \equiv \mathbf{At}(t, \textsc{Boiling}(e))))))$$

Finally, let us define $\textsc{BoilingFor}(\theta_e^i, \theta_l^t, \theta_h^t)$ to stand for the claim that θ_e^i has been boiling for between θ_l^t and θ_h^t units, and was uncooked before that:

Definition 5.3.6

$$\textsc{BoilingFor}(\theta_e^i, \theta_l^t, \theta_h^t) \overset{\text{def}}{=}$$
$$\exists v_s^t(\mathbf{now} - \theta_h^t \leq v_s^t \leq \mathbf{now} - \theta_l^t \wedge \mathbf{At}(v_s^t, \textsc{UnCooked}(\theta_e^i)) \wedge$$
$$\forall v^t(v_s^t < v^t \leq \mathbf{now} \supset \mathbf{At}(v^t, \textsc{Boiling}(\theta_e^i)),$$
where v_s^t and v^t do not occur free in θ_e^i, θ_l^t, and θ_h^t

We can now sketch how a proof of our main result (conjecture 5.3.13) would go. Let's first look at $\textsc{startBoil}$, the initial action. We should be able to show that given that the agent initially knows of a clock c that it is accurate and of an egg e that it is uncooked, he must be able to achieve the goal of knowing that c is still accurate and that e has been boiling for 0 time units by doing the action of starting to boil e:

Conjecture 5.3.14

$$\models \forall e \forall c(\mathbf{Know}(\textsc{Accurate}(c) \wedge \textsc{UnCooked}(e)) \supset$$
$$\mathbf{Can}(\textsc{startBoil}(e), \mathbf{Know}(\textsc{Accurate}(c) \wedge \textsc{BoilingFor}(e, 0, 0))))$$

Now let us consider the $\textsc{lookAt}$ action. From the assumptions, it should be possible to prove the following conjecture, which says that if the agent knows of clock c that it is accurate and of egg e that it has been boiling for between l and h time units, then by looking at c, he is able to achieve the goal of knowing what time c indicates and knowing that c is still accurate and that e has been boiling for between $l+1$ and $h+10$ time units (since $\textsc{lookAt}$ must have taken between 1 and 10 units):

Conjecture 5.3.15

$$\models \forall l \forall h \forall e \forall c(\mathbf{Know}(\textsc{Accurate}(c) \wedge \textsc{BoilingFor}(e, l, h)) \supset$$
$$\mathbf{Can}(\textsc{lookAt}(c), \exists t_c \mathbf{Know}(\textsc{value}(c) = t_c \wedge \textsc{Accurate}(c) \wedge$$
$$\textsc{BoilingFor}(e, l+1, h+10))))$$

By instantiating this, we get the following claim, which applies to the $\textsc{lookAt}$ done following the initial $\textsc{startBoil}$:

Conjecture 5.3.16

$$\models \forall e \forall c(\mathbf{Know}(\textsc{Accurate}(c) \wedge \textsc{BoilingFor}(e, 0, 0)) \supset$$
$$\mathbf{Can}(\textsc{lookAt}(c), \exists t_c \mathbf{Know}(\textsc{value}(c) = t_c \wedge \textsc{Accurate}(c) \wedge \textsc{BoilingFor}(e, 1, 10))))$$

Now for the WAIT action, we should be able to prove the following conjecture, which says that if the agent knows of clock c that it is accurate and of egg e that it has been boiling for between l and h time units, then by doing WAIT, he is able to achieve the goal of knowing that c is still accurate and that e has been boiling for between $l+1$ and $h+10$ time units:

Conjecture 5.3.17

$$\models \forall l \forall h \forall e \forall c(\mathbf{Know}(\textsc{Accurate}(c) \wedge \textsc{BoilingFor}(e,l,h)) \supset \\ \mathbf{Can}(\textsc{wait}, \mathbf{Know}(\textsc{Accurate}(c) \wedge \textsc{BoilingFor}(e, l+1, h+10))))$$

Then by induction on k, we should be able show that if the agent knows what time clock c indicates, knows of c that it is accurate, and knows of egg e that it has been boiling for between l and h time units, then by repeatedly waiting and looking at c (up to k times) for as long as 500 time units have not elapsed, he is able to achieve the goal of knowing what time c indicates and knowing that c is still accurate and that e has been boiling for a length of time that is at least as large as the least of $l+2\mathrm{k}$ and 500 units and at most as large as the least of $h+20\mathrm{k}$ and 520 units:

Conjecture 5.3.18

For all $\mathrm{k} \in \mathbb{N}$,

$$\models \forall l \forall h \forall e \forall c(\exists t_c \mathbf{Know}(\textsc{value}(c) = t_c \wedge \textsc{Accurate}(c) \wedge \textsc{BoilingFor}(e,l,h)) \supset \\ \mathbf{Can}(\textsc{wait500u}_{\mathrm{k},0}, \exists t_c \mathbf{Know}(\textsc{value}(c) = t_c \wedge \textsc{Accurate}(c) \wedge \\ \textsc{BoilingFor}(e, \textsc{min}(l+2\mathrm{k}, 500), \textsc{min}(h+20\mathrm{k}, 520)))))$$

By instantiating this with the values used in our example, we obtain that if the agent knows what time clock c indicates, knows of c that it is accurate, and knows of egg e that it has been boiling for between 1 and 10 units, then by repeatedly waiting and looking at c (up to 250 times) for as long as 500 time units have not elapsed, he is able to achieve the goal of knowing what time c indicates and knowing that c is still accurate and that e has been boiling for between 500 and 520 units:

Conjecture 5.3.19

$$\models \forall e \forall c(\exists t_c \mathbf{Know}(\textsc{value}(c) = t_c \wedge \textsc{Accurate}(c) \wedge \textsc{BoilingFor}(e,1,10)) \supset \\ \mathbf{Can}(\textsc{wait500u}_{250,0}, \exists t_c \mathbf{Know}(\textsc{value}(c) = t_c \wedge \textsc{Accurate}(c) \wedge \\ \textsc{BoilingFor}(e, 500, 520))))$$

Finally, for the concluding action $\textsc{stopBoil}$, we should be able to show that if the agent knows what time clock c indicates, knows of c that it is accurate, and knows of egg e that it has been boiling for between 500 and 520 units, then by stopping e from boiling, the agent is able to achieve the main goal of having soft-boiled e:[5]

Conjecture 5.3.20

$$\models \forall e \forall c(\exists t_c \mathbf{Know}(\textsc{value}(c) = t_c \wedge \textsc{Accurate}(c) \wedge \textsc{BoilingFor}(e, 500, 520)) \supset \mathbf{Can}(\textsc{stopBoil}(e), \textsc{SoftBoiled}(e)))$$

Then by chaining conjectures 5.3.14, 5.3.16, 5.3.19, and 5.3.20, it should be easy to prove the main conjecture, which says that if the agent knows of a clock c that it is accurate and of an egg e that it is uncooked, then he is able to achieve the goal of having soft-boiled e by first putting e to boil, then looking at c, then repeatedly waiting and looking at c (up to k times) for as long as 500 time units have not elapsed, and then removing e from boiling:

Conjecture 5.3.13

$$\models \forall e \forall c(\mathbf{Know}(\textsc{Accurate}(c) \wedge \textsc{UnCooked}(e)) \supset \mathbf{Can}((\textsc{startBoil}(e); \textsc{lookAt}(c); \textsc{wait500u}_{250,0}; \textsc{stopBoil}(e)), \textsc{SoftBoiled}(e)))$$

5.3.4 Frying Fish Until Flesh Is Opaque

Our last temporal example is a case where one need not even know how long some process needs to be left going in order to achieve a desired state, as one can simply look for signs that the process has gone on for long enough in the situation itself. This is analogous to the scanning example in chapter 4, where one did not even need to know the relative position of the target object. The example is based on a common cooking technique used to determine if a fish filet has been frying for the right amount of time: one simply keeps looking at the flesh of the fish and once it stops being "transparent", the fish is ready and should be taken off the heat. Using this technique, it is possible to avoid overcooking or undercooking the fish. It is often easier and more effective to do this kind of monitoring of the situation rather than to attempt to calculate the required duration of the process in advance (which would often depend on the details of the situation, for example, the exact size of the fish)

[5]In fact, it should be possible to show that in these conditions the agent is able to achieve the stronger goal $\textsc{JustSoftBoiled}(e)$; this also applies to the main conjecture (5.3.13).

and timing its progression.

The conjecture below is a formal version of the example. It says that if the agent knows that a fish f is uncooked, then he is able to achieve the goal that f be ready by first putting it to fry, then looking at it, then repeatedly waiting and looking at it again for as long as its flesh is still transparent, and then removing it from frying (the number of iterations being at most 250).

Conjecture 5.3.21

$$\models \forall f(\mathbf{Know}(\textsc{UnCooked}(f)) \supset$$
$$\mathbf{Can}((\textsc{startFry}(f); \textsc{lookAt}(f); \mathbf{while}_{250}(\textsc{Transpa}(f), (\textsc{wait}; \textsc{lookAt}(f)));$$
$$\textsc{stopFry}(f)), \textsc{Ready}(f)))$$

We will not develop a formalization of this situation, but the specification of the actions involved should be straightforward and a proof for conjecture 5.3.21 should be similar in structure to the ones developed in the last two sections.

5.4 De Re Knowledge and its Role in Action

It should be possible to exploit the insights gained as we apply our theory to obtain a better understanding of the various notions of knowing-who and *de re* knowledge and of what role they play in action. As we pointed out earlier, the standard-name account of these notions is in many cases unsatisfactory. Moreover, there seems to be another notion of *de re* knowledge that has more to do with having a certain kind of causal connection with the object; for example, if I see the person by the punch bowl, there seems to be a sense in which my perceptual state is of that person even though I need not know who she is. This notion of *de re* seems to have mainly to do with knowing how to act upon the object (even perception is an activity). For an agent like the robot of the previous section, it would seems quite appropriate to take knowing what an object is in this "causal contact" sense to amount to knowing how to get to the object's location. If this view is correct, then given a particular type of agent and its action repertory, it may be possible to formally reduce this notion of *de re* to that of ability to act upon the object (perhaps in a restricted set of ways), where ability would involve only knowledge of primitive actions and other entities whose characterization as mental representations is unproblematic, such as relative locations, joint rotation angles, phone numbers, etc. Then it may be possible to develop an account of actions with object parameters where ability to do the action follows automatically from the ability to act upon the object. This would mean, for example, that the action of going where an object is and picking it up discussed in section 4.7 would not need to be explicitly defined; it would follow from the fact that an agent must know how to get to an object in order to "know what object it is" that the agent can do the action if he "knows what the object is".

In simple domains such as that of our robot, this analysis of *de re* seems quite appealing. But, the approach has some counterintuitive consequences: there is no requirement to even appeal to knowledge of relatively high-level entities such as relative locations; ability to do high-level actions, such as sending an electronic message to someone giving a talk at a remote institution, can be reduced to knowledge of what primitive actions to do in what circumstances, such as knowing what finger movements to do. But a real person would be prepared to deal with a huge number of unforeseen eventualities arising during the execution of his plan to send a message; he might have to track down the speaker who has moved, etc. It seems that for agents with open-ended action repertories, the flexibility and robustness of behavior cannot be explained without appealing to knowledge of at least some of the objects acted upon, for example, without the agent at some point knowing who the speaker was. The reduction does not capture the right generalizations. But if some other kind of *de re* knowledge is required in these cases, what does it really amount to and how is it related to the "causal contact" notion, which seems equally required for action? Perhaps what is needed is some kind of notion of internal identification; a kind of relaxed version of "having a standard name" that acknowledges the possibility of various forms of misidentification.

5.5 Conclusion

The examples discussed in the last two chapters show that the distinction between indexical and objective knowledge is useful and applicable across a whole range of domains. Some of these examples, in particular the ones about data structure traversal, also show that the notions involved are quite abstract and not tied to physical space.

The examples dealing with temporally constrained goals have a significance beyond merely showing that the distinction between indexical and objective knowledge is applicable to the temporal domain; they also indicate that the distinction is an essential one, one of great practical importance, one that cannot be mimicked or defined away. The other examples involve mostly spatial indexicality, the distinction between knowing where something is relative to oneself and knowing where it is in absolute terms. As mentioned in section 1.1, one could argue that it is possible to handle these spatial cases within frameworks, such as Moore's, that are simpler than the one developed here: one should treat the indexical cases, that is, the ones involving knowledge about where something is relative to **here**, as *de re* cases, that is, as cases involving the agent knowing *of* himself that something is at such and such position relative to that agent; this works, of course, only if one assumes that agents always know who they are; but in many domains, this assumption seems reasonable.

However, to define temporally indexical knowledge in terms of *de re* knowledge of times in a similar fashion would amount to ignoring the distinction between indexical and absolute temporal knowledge. This distinction is clearly required by our temporal examples. One certainly cannot assume that agents always know what time it is. One needs to capture the fact that even if one knows that φ holds at absolute time v^t and knows that if φ currently holds, then φ' must also currently hold, and v^t happens to be the current time, one need not know that φ' currently holds:

$$\not\models \exists v^t(v^t = \mathbf{now} \wedge \mathbf{Know}(\mathbf{At}(v^t, \varphi))) \wedge \mathbf{Know}(\varphi \supset \varphi') \supset \mathbf{Know}(\varphi')$$

But if one also knows what time it is, then one must know that φ' currently holds:

$$\models \exists v^t(\mathbf{Know}(\mathbf{At}(v^t, \varphi)) \wedge \mathbf{Know}(v^t = \mathbf{now})) \wedge \mathbf{Know}(\varphi \supset \varphi') \supset \mathbf{Know}(\varphi')$$

now appears to behave much like an ordinary non–rigid constant, but it cannot just be treated as such. It has many properties that ordinary temporal constants do not have; for example, the positive introspection "axiom" $\models \mathbf{Know}(\varphi) \supset \mathbf{Know}(\mathbf{At}(\mathbf{now}, \mathbf{Know}(\varphi)))$ (proposition 3.14); this is no longer valid if **now** is replaced by an arbitrary temporal constant. Other examples are the proposition that says that agents know what primitive actions they have just done (proposition 3.75) and the "axiom" of specialization (proposition 3.3). A framework where these properties did not hold could not claim to capture the logic of temporally indexical knowledge. We fail to see how these properties could be captured without using a language and a semantics that goes very much along the lines of our proposal.

Note that we are not claiming that the misguided strategy of translating temporally indexical knowledge to *de re* temporal knowledge is attractive from the point of view of Moore's framework; it would be a more likely suggestion if one had a logic of temporal knowledge that dealt only with absolute time.[6] As discussed in sections 2.1.1 and 3.7, the problem with Moore's framework is not so much with temporally indexical knowledge as with knowledge of absolute times. Moore's semantics is based on a domain of states, rather than separate domains of worlds and times. So he can only talk about what holds in the current state or in the states that would result from doing an action. His framework is not a true logic of time, but only a logic of action.

It would be nice to have analogous examples to show that the distinction between knowing and ignoring who one is also is of more than philosophical significance (the usual amnesia examples are not too convincing in this regard). For agents that are much simpler than humans, it would not seem all that unusual to have them not know who they are; in fact, one is hard pressed to come up with good reasons for them to need to know who they are. Perhaps the area of distributed systems and communication protocols would be a good place to look for cases where the distinction is really useful. The work of Grove and Halpern (1989) seems to suggest that it is.

The issues raised in the previous section as well as the difficulties encountered with the action GOWHERE in section 4.7 also show that many questions remain with respect to the nature of *de re* knowledge and the role it needs to play in action. Let us conclude by claiming that the examples we have presented provide convincing evidence that our framework allows a much more accurate modeling of the knowledge prerequisites and effects of actions than frameworks that ignore indexicality.

[6]We sketched what the semantics of such a logic would look like at the beginning of section 3.5.7.

Chapter 6

Conclusion

Agents act upon and perceive the world from a particular perspective. It is important to recognize this relativity to perspective if one is not to be overly demanding in specifying what they need to know in order to be able to achieve goals through action. They need not know much about their objective situation; what they need is indexical knowledge, knowledge about how they are related to things in their environment or to events in their history. And perception yields just the kind of indexical knowledge that is needed for action.

Previous formal accounts of the ability of agents to achieve goals by doing actions, such as that of Moore (1980; 1985) and Morgenstern (1987; 1988), have ignored this, and thus end up imposing knowledge requirements that are neither necessary nor sufficient for ability; they fail to properly specify the knowledge prerequisites and effects of actions. In this thesis, we have developed a formal solution to the problem of indexical knowledge prerequisites and effects of action.

6.1 Contributions

In this section, we discuss the ways in which our theory improves over previous accounts of the relationship between knowledge and action.

Our account of knowledge is the first to formally capture the distinction between indexical and *de re* knowledge: it allows an agent to know something in an indexical way without knowing it *de re* and vice-versa. Also, the account fully handles time, in particular, temporally indexical knowledge. It has a simple syntax which makes it easy to make assertions involving indexical knowledge. Its model-theoretic semantics is a simple and natural extension of standard possible-world semantic schemes for the logic of knowledge. It retains what is perhaps the most important feature of possible-world semantics: the correspondence between constraints on the accessibility relation and important properties of the notion formalized. Finally, our account includes a formalization of the requirement that knowledge be persistent that works well with both temporally indexical and temporally absolute knowledge.

Note also that the temporal part of our logic is simple and very expressive. Both eternal and indexical temporal assertions can be made. Relations between indexically specified and objectively specified times can also be expressed. The fact that time is reified allows quantification over times into epistemic contexts, so one can make very weak ascriptions of temporal knowledge; for example, one can say that an agent knows that another agent knows what time it is without himself knowing it. The occurrence of concurrent actions, continuous processes, and actions involving several agents can be expressed.

Our formalization of ability improves over previous accounts in several ways. Firstly, it does not require an agent to know who he is in order to be able to achieve a goal by doing an action. All references to the agent within the knowledge requireed are indexical references. For instance, in the simple action case we require the agent to know that it is physically possible for *himself* to do the action and that if *he himself* does the action, the goal will necessarily hold afterwards (as well as knowing what the action is). Mere *de re* knowledge is neither necessary nor sufficient for the agent to be able to achieve the goal (a concrete example of this was given in chapter 4). Situations where an agent does not know who he is (in the sense of not knowing an objective standard name for himself) are perhaps unusual, but our theory could not claim to reflect a real understanding of the phenomenon of indexicality if it did not deal with such cases.

Secondly, our account of ability is based on a very expressive temporal logic. We can thus handle prerequisites or effects of actions that involve knowledge of absolute times and knowing what time it is, as well as many cases of actions that refer to times (for example, the action of locking up at 5 p.m., which is formalized in section 5.3). This would also make it easier to extend the formalization to handle more complex actions than are currently treated, for instance concurrent actions, actions involving several agents, and actions that refer to time in very general ways.

Finally, the underlying logic includes an adequate treatment of indexical knowledge in general, which permits a more accurate specification of the knowledge prerequisites and effects of actions.

We have then applied the framework developed to the formalization of various domains where agents have goals to achieve and certain repertories of actions to do so. We have shown how the resulting theory allows us to prove that the agent is able to achieve certain goals if he knows certain facts. We have also shown how the actions involved should be formalized so as to avoid making excessive requirements upon the knowledge of agents — so that only indexical knowledge, as opposed to *de re* knowledge, is required when that is sufficient.

We first examined applications of the theory in the robotics domain, where indexicality plays a particularly important role. We have shown how actions can be formalized, given that perception yields indexical knowledge, and that ability to act upon an object does not require *de re* knowledge of the object or its absolute position. We have also shown how indexical knowledge and objective knowledge can be related in our framework to deal with the use of maps for navigation. We have discussed the representational issues that arise, which have general relevance to the formalization of actions with indexical knowledge prerequisites or effects. Finally, we discussed problems that arise in handling certain higher-level parametrized actions and proposed a solution.

We then developed applications of the theory in several other domains, which show that the distinction between indexical and objective knowledge is generally useful and more abstract than initial impressions might suggest, in particular, that it is in no way tied to the notion of physical space. An example involving an agent who wants to make a phone call was first formalized; we showed how this could be handled in a way that reflects the fact that the agent need not always know what his area code is. We next looked at a very abstract domain, that of data structure search and manipulation; it was shown how indexical knowledge was also relevant in this case. Three situations where temporal knowledge is required for ability were next examined. We found that the distinctions between various kinds of knowledge that proved useful in spatial contexts are similarly useful in dealing with ability in temporal contexts. Various problems with the notion of *de re* knowledge and its role in action were then discussed. We concluded by arguing that the applications developed, in particular the ones involving temporal knowledge, provide convincing evidence that the distinction between indexical and objective knowledge supported by our framework has substantial practical value and that it cannot be done justice within existing accounts of ability.

6.2 Directions for Future Research

We now conclude by discussing various ways in which this work could be extended or applied to other areas.

Situations involving temporally constrained goals and actions that refer to times, like those discussed in section 5.3, play a major role in motivating the distinction between indexical and objective knowledge that is central to our theory. They show that there are practically significant cases that cannot be handled without accommodating this distinction at a primitive level. But the examples discussed in that section do not nearly exhaust the kinds of situations involving temporal knowledge that would be worth examining. For instance, making use of a schedule raises the same kind of issues as that involved in the use of maps in spatial domains. Moreover, the way the examples are treated is rather tentative; it should be possible to develop more elegant and general treatments. Insights as to how this could be done might be obtainable from existing work on programming languages targeted at real-time applications and verification techniques applicable to such languages. The work of Davis (1989a) on the formalization of actions involving the monitoring of the agent's environment for the occurrence of perceivable conditions should also be looked at.

Also, as pointed out in section 5.5, it would be desirable to obtain examples of commonly occurring situations where the distinction between knowing and ignoring who one is proves useful; this would provide additional motivation for our framework. The work of Grove and Halpern (1989) suggests that the area of distributed systems may be a good place to look for such examples.

Many questions remain concerning the nature of *de re* knowledge and the role it plays in action, as the discussion in sections 5.4 and 4.7 makes clear. The issues involved in this should be examined further; we believe that the theory developed in this thesis and the examples discussed should provide new insights into these issues and could perhaps lead to a better account of *de re* knowledge.

Our account of ability should also be extended to handle more complex types of actions, such as those involving indefinite iteration and concurrency, as well as plans involving multiple agents. Such cases are handled by Morgenstern (1987; 1988) and Nunes (1990), but their accounts do not deal with indexicality. Situations involving interacting agents would be especially interesting from the indexicality point of view, since the difference in perspective between agents must be accounted for if they are to have indexical knowledge of the same facts. For example, if we were cooperating in the dismantling of a motor, and were facing each other, and if we needed to agree that a particular wrench was to be used next, I would need to know that it was the wrench on my left and you would need to know that it was the wrench on your right.

It would also be desirable to have a mathematically satisfactory axiomatization of the logic — one that is complete, if this is achievable. This might require a re-examination of some aspects of the logic.

One should also examine how a planning program based on the theory could be implemented, one that can handle situations where agents have only indexical (and incomplete) knowledge. Various approaches to the development of planners based on theories of ability have been tried. Moore (1980) formalizes the semantics of his theory within first-order logic and then uses a normal theorem prover to reason with it; but this yields a planner that is too inefficient to be applied to realistic situations. Morgenstern (1988) discusses work by Meth and Davis directed at developing a relatively efficient STRIPS-style planner that is sound but not complete with respect to her theory.

As it is, however, our theory is far from realistic in the reasoning power it ascribes to agents: it assumes that agents know all logical consequences of what they know; it is also subject to the frame problem. This limits its usefulness as a specification for a planner. One should look for principled ways of restricting the inferential power of agents in the theory so that in well defined classes of situations, the planning problem is tractable or at least decidable.

There are two application areas where the theory developed in this thesis has the potential for yielding very significant benefits. The first is that of discourse processing, in particular, formal models of linguistic communication. Some existing models of language use, such as that developed by Cohen and Levesque (1990), include sophisticated accounts of how the knowledge and intentions of agents are involved in speech acts, but these accounts generally ignore indexicality, both in utterances and mental attitudes. Other less computationally oriented models, such as that of Barwise and Perry (1983) and Katagiri (1989), include elaborate treatments of indexicality, but have little to say about what happens after one has obtained a representation of the interpretation of the utterance; they do not address the issue of how the agent can draw inferences from these indexical representations and his prior knowledge, and ultimately take action. An integrated account of reasoning and action, both linguistic and non-linguistic, that accommodates indexicality is needed. Our theory would provide a good foundation for such an account. Some of the key issues that would have to be addressed are how one should account for the perspective shift that occurs as information is communicated by one agent to another, and how one should deal with indexicals like 'you' and demonstratives like 'this person', which do not have obvious definitions in terms of **self** and **now**. It would also be interesting to see if the so called "self-referentiality" of speech acts could not be treated as a case of indexicality. Speech acts are "self-referential" in the sense that the speaker must utter the sentence with a particular intention, intention that refers to the act itself in its specification of how the ultimate intended effects are to be achieved, that is, through the addressee's understanding of the act itself.

The other application area for which this theory would appear to hold much promise is that of logical formalisms for the design of reactive agents, such as autonomous robots. In many domains, it is essential that agents be able to respond in a timely manner to certain environmental conditions (e.g., to avoid colliding with a moving object). As mentioned in section 2.3, architectures have been proposed for such reactive agents that emphasize environment monitoring and the selection of actions appropriate to conditions; the focus is on plan execution as opposed to plan generation. Rosenschein and Kaelbling (1986) have developed a framework for the design of reactive agents that involves specifying the desired behavior in a logical language and using tools to compile the specification into a machine-level description that satisfies it; plan generation occurs not at run-time but at compile-time (Kaelbling and Rosenschein 1990). Moreover, it has been argued repeatedly that indexical representations play a major role in the production of reactive behavior (Agre and Chapman 1987; Subramanian and Woodfill 1989). Thus, it would appear that aspects of our logic could prove useful as part of a formalism for the design of reactive agents. Also, studying the role of indexical representations in such agents could yield further insights into indexicality and suggest refinements to our theory.

Bibliography

Agre, P. E. 1990. Review of (Suchman 1987). *Artificial Intelligence*, 43(3):369–384.

Agre, P. E. and Chapman, D. 1987. Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 268–272, Seattle, WA. American Association for Artificial Intelligence, Morgan Kaufmann Publishing.

Agre, P. E. and Chapman, D. 1990. What are plans for? *Robotics and Autonomous Systems*, 6:17–34.

Aho, A. V., Hopcroft, J. E., and Ullman, J. D. 1974. *The Design and Analysis of Computer Algorithms.* Addison-Wesley Publishing, Reading, Mass.

Allen, J. F. 1984. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154.

Barwise, J. 1986. Information and circumstance. *Notre Dame Journal of Formal Logic*, 27(3):324–338.

Barwise, J. and Perry, J. 1983. *Situations and Attitudes.* MIT Press/Bradford Books, Cambridge, Mass.

Castañeda, H.-N. 1968. On the logic of attributions of self-knowledge to others. *Journal of Philosophy*, 65(15):439–456.

Chapman, D. 1987. Planning for conjunctive goals. *Artificial Intelligence*, 32:333–377.

Chellas, B. F. 1980. *Modal Logic: An Introduction.* Cambridge University Press.

Cohen, P. R. and Levesque, H. J. 1987. Persistence, intention, and commitment. Technical Report 415, AI Center, SRI International, Menlo Park, CA.

Cohen, P. R. and Levesque, H. J. 1990. Rational interaction as the basis for communication. In Cohen, P. R., Morgan, J., and Pollack, M. E., editors, *Intentions in Communication*. MIT Press, Cambridge, MA.

Davis, E. 1988. Inferring ignorance from the locality of visual perception. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 786–791, St. Paul, MN. American Association for Artificial Intelligence.

Davis, E. 1989a. Reasoning about hand-eye coordination. In *IJCAI Workshop on Knowledge, Perception, and Planning*, Detroit.

Davis, E. 1989b. Solutions to a paradox of perception with limited acuity. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 79–82, Toronto, Ont. Morgan Kaufmann Publishers.

Fikes, R. and Nilsson, N. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208.

Georgeff, M. P. 1987. Planning. *Annual Review of Computer Science*, 2:359–400.

Goldblatt, R. 1987. *Logics of Time and Computation*. CSLI Lecture Notes No. 7. Center for the Study of Language and Information, Stanford University, Stanford, CA.

Grove, A. J. and Halpern, J. Y. 1989. Naming and anonymity in a multi-agent epistemic logic. Unpublished manuscript, Department of Computer Science, Stanford University.

Haas, A. R. 1986. A syntactic theory of belief and action. *Artificial Intelligence*, 28:245–292.

Halpern, J. Y. and Moses, Y. 1985. A guide to the modal logics of knowledge and belief: Preliminary draft. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 480–490, Los Angeles. Morgan Kaufmann Publishing.

Hintikka, J. 1962. *Knowledge and Belief*. Cornell University Press, Ithaca, NY.

Hintikka, J. 1969. Semantics for the propositional attitudes. In Davis, J. W., Hockney, D. J., and Wilson, K. W., editors, *Philosophical Logic*. D. Reidel Publishing, Dordrecht, Holland.

Hughes, G. E. and Cresswell, M. J. 1968. *An Introduction to Modal Logic.* Methuen, London, UK.

Israel, D. J. 1987. The role of propositional objects of belief in action. Technical Report CSLI-87-72, CSLI, Stanford University, Stanford, CA.

Kaelbling, L. P. and Rosenschein, S. J. 1990. Action and planning in embedded agents. *Robotics and Autonomous Systems*, 6:35–48.

Kaplan, D. 1969. Quantifying in. In Davidson, D. and Hintikka, J., editors, *Word and Objections: Essays on the Work of W.V.O. Quine*, pages 206–242. D. Reidel Publishing, Dordrecht, Holland.

Kaplan, D. 1975. How to Russell a Frege-Church. *Journal of Philosophy*, 72:716–729.

Kaplan, D. 1977. Demonstratives. Unpublished manuscript, UCLA.

Katagiri, Y. 1989. Semantics of perspectival utterances. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1474–1479, Detroit. Morgan Kaufmann Publishing.

Konolige, K. 1982. A first order formalization of knowledge and action for a multi-agent planning system. In Hays, J. and Michie, D., editors, *Machine Intelligence*, volume 10. Ellis Horwood, Chichester, UK.

Konolige, K. 1986. *A Deduction Model of Belief.* Pitman Publishing.

Kripke, S. A. 1963. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94.

Lakemeyer, G. 1990. *Models of Belief for Decidable Reasoning in Incomplete Knowledge Bases.* PhD thesis, Department of Computer Science, University of Toronto.

Lespérance, Y. 1989. A formal account of self-knowledge and action. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 868–874, Detroit. Morgan Kaufmann Publishing.

Lespérance, Y. and Levesque, H. J. 1990. Indexical knowledge in robot plans. In *Proceedings of the Eight National Conference on Artificial Intelligence*, pages 868–874, Boston. American Association for Artificial Intelligence, AAAI Press / The MIT Press.

Levesque, H. J. 1984a. Foundations of a functional approach to knowledge representation. *Artificial Intelligence*, 23:155–212.

Levesque, H. J. 1984b. A logic of implicit and explicit belief. In *Proceedings of the National Conference on Artificial Intelligence*, pages 198–202, Austin, TX. American Association for Artificial Intelligence.

Lewis, D. 1979. Attitudes *de dicto* and *de se*. *The Philosophical Review*, 88(4):513–543.

Lifschitz, V. 1987. On the semantics of strips. In Georgeff, M. P. and Lansky, A. L., editors, *Reasoning about Actions and Plans*, pages 1–9, Los Altos, CA. Morgan Kaufmann Publishing.

Lipson, J. D. 1981. *Elements of Algebra and Algebraic Computing.* Addison-Wesley Publishing, Reading, Mass.

McArthur, G. L. 1988. Reasoning about knowledge and belief: a survey. *Computational Intelligence*, 4(3):223–243.

McCarthy, J. and Hayes, P. 1979. Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B. and Michie, D., editors, *Machine Intelligence*, volume 4, pages 463–502. Edinburgh University Press, Edinburgh.

McDermott, D. V. 1982. A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6:101–155.

Mendelson, E. 1979. *Introduction to Mathematical Logic (Second Edition).* D. Van Nostrand Co., New York.

Moore, R. C. 1980. Reasoning about knowledge and action. Technical Report 191, AI Center, SRI International, Menlo Park, CA.

Moore, R. C. 1985. A formal theory of knowledge and action. In Hobbs, J. R. and Moore, R. C., editors, *Formal Theories of the Common Sense World.* Ablex Publishing, Norwood, NJ.

Morgenstern, L. 1987. Knowledge preconditions for actions and plans. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 867–874, Milan. Morgan Kaufmann Publishing.

Morgenstern, L. 1988. *Foundations of a Logic of Knowledge, Action, and Communication.* PhD thesis, Department of Computer Science, New York University.

Nunes, J. H. T. 1990. Ability and commitment. Unpublished manuscript, Department of Computer Science, University of Toronto.

Perry, J. 1977. Frege on demonstratives. *Philosophical Review*, 86(4):474–497.

Perry, J. 1979. The problem of the essential indexical. *Noûs*, 13:3–21.

Quine, W. V. O. 1956. Quantifiers and the propositional attitudes. *Journal of Philosophy*, 53.

Quine, W. V. O. 1971. Reference and modality. In Linsky, L., editor, *Reference and Modality*, pages 17–33. Oxford University Press, London.

Rapaport, W. J. 1986. Logical foundations for belief representation. *Cognitive Science*, 10:371–422.

Rescher, N. and Urquhart, A. 1971. *Temporal Logic.* Springer-Verlag, Vienna.

Richard, M. 1983. Direct reference and ascriptions of belief. *Journal of Philosophical Logic*, 12:425–452.

Rosenschein, S. J. and Kaelbling, L. P. 1986. The synthesis of digital machines with provable epistemic properties. In Halpern, J. Y., editor, *Theoretical Aspects of Reasoning about Knowledge: Proceedings of the 1986 Conference*, pages 83–98, Monterey, CA. Morgan Kaufmann Publishing.

Sacerdoti, E. 1977. *A Structure for Plans and Behavior.* Elsevier, North Holland, New York.

Shoham, Y. 1987. Temporal logics in AI: Semantical and ontological considerations. *Artificial Intelligence*, 33(1):89–104.

Smith, B. C. 1986. Varieties of self-reference. In Halpern, J. Y., editor, *Theoretical Aspects of Reasoning about Knowledge: Proceedings of the 1986 Conference*, pages 19–43, Monterey, CA. Morgan Kaufmann Publishing.

Stalnaker, R. C. 1981. Indexical belief. *Synthese*, 49:129–151.

Subramanian, D. and Woodfill, J. 1989. Making the situation calculus indexical. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 467–474, Toronto, Canada. Morgan Kaufmann Publishing.

Suchman, L. A. 1987. *Plans and Situated Action: The Problem of Human–Machine Communication.* Cambridge University Press, Cambridge, England.

Thomason, R. C. 1984. Combinations of tense and modality. In Gabbay, D. and Guenthner, F., editors, *Handbook of Philosophical Logic*, volume 2, pages 135–165. D. Reidel Publishing, Dordrecht, Holland.

Formal Syntax of the Logic

Syntactic categories

Category name	Syntactic variable	Membership
Agent variables	v^a	$\{a_0, a_1, \ldots\}$
Individual variables	v^i	$\{i_0, i_1, \ldots\}$
Temporal variables	v^t	$\{t_0, t_1, \ldots\}$
Action variables	v^d	$\{d_0, d_1, \ldots\}$
Agent functions	$\boldsymbol{f}^a$	$\{\mathtt{F}^a_0, \mathtt{F}^a_1, \ldots\}$
Individual functions	$\boldsymbol{f}^i$	$\{\mathtt{F}^i_0, \mathtt{F}^i_1, \ldots\}$
Temporal functions	$\boldsymbol{f}^t$	$\{\mathtt{F}^t_0, \mathtt{T}^i_1, \ldots\}$
Action functions	$\boldsymbol{f}^d$	$\{\mathtt{F}^d_0, \mathtt{F}^d_1, \ldots\}$
Predicate symbols	$\boldsymbol{R}$	$\{\mathrm{R}_0, \mathrm{R}_1, \ldots\}$
Variables	v	
Agent terms	θ^a	
Individual terms	θ^i	
Temporal terms	θ^t	
Action terms	θ^d	
Formulas	φ	

Formation rules

$$
\begin{array}{rcl}
v & ::= & v^a \mid v^i \mid v^t \mid v^d \\
\theta^a & ::= & v^a \mid \mathbf{self} \mid \boldsymbol{f}^a(\theta^i_1, \ldots, \theta^i_n) \\
\theta^i & ::= & \theta^a \mid v^i \mid \boldsymbol{f}^i(\theta^i_1, \ldots, \theta^i_n) \\
\theta^t & ::= & v^t \mid \mathbf{now} \mid \boldsymbol{f}^i(\theta^{i|t}_1, \ldots, \theta^{i|t}_n) \\
\theta^d & ::= & v^d \mid \boldsymbol{f}^d(\theta^i_1, \ldots, \theta^i_n) \\
\varphi & ::= & \boldsymbol{R}(\theta^i_1, \ldots, \theta^i_n) \mid \mathbf{Does}(\theta^d, \theta^t) \mid (\theta^a_1 = \theta^a_2) \mid (\theta^i_1 = \theta^i_2) \mid (\theta^t_1 = \theta^t_2) \mid (\theta^d_1 = \theta^d_2) \mid (\theta^t_1 < \theta^t_2) \mid \\
 & & \neg\varphi \mid (\varphi_1 \supset \varphi_2) \mid \forall v \varphi \mid \mathbf{At}(\theta^t, \varphi) \mid \mathbf{By}(\theta^a, \varphi) \mid \mathbf{Know}(\varphi) \mid \Box\varphi
\end{array}
$$

Glossary

Primitive Symbols

Symbol	Meaning
self	the agent of the context
now	the time of the context
$(\theta^t_1 < \theta^t_2)$	θ^t_1 is earlier than θ^t_2
Does(θ^d, θ^t)	**self** does action θ^d from **now** to time θ^t
At(θ^t, φ)	φ holds at time θ^t
By(θ^a, φ)	agent θ^a has property φ
Know(φ)	**self** knows that φ
$\Box\varphi$	φ is historically necessary

Action Operators

Definitions on page 58

Expression	Meaning
skip	doing nothing
$(\delta_1; \delta_2)$	doing δ_1 and δ_2 in sequence
if$(\varphi, \delta_1, \delta_2)$	doing δ_1 if φ holds and δ_2 otherwise
δ^{k}	doing δ k times in sequence
ifThen(φ, δ)	doing δ if φ holds and nothing otherwise
while$_{\mathsf{k}}(\varphi, \delta)$	doing δ up to k times as long as φ remains true

Glossary

Derived Operators

Formula	Meaning	Def.
$\Diamond\varphi$	φ is historically possible	57
Know(θ^a, φ)	θ^a knows that φ	57
and similarly for **Does**(θ^a, δ), **Can**$(\theta^a, \delta, \varphi)$, etc.		
DoesNext(δ)	**self** does action δ from **now** to some future time	59
DoneFromTo$(\delta, \theta^t_s, \theta^t_e)$	**self** does δ from time θ^t_s to time θ^t_e	59
DoneFrom(δ, θ^t)	**self** does δ from time θ^t to **now**	59
Done(δ)	**self** does δ from some past time to **now**	59
PhyPoss(δ)	it is "physically possible" for **self** to do action δ next	60
AfterNec(δ, φ)	if **self** does δ next, then φ must hold afterwards	60
Res(δ, φ)	δ results in φ	60
Kwhether(φ)	**self** knows whether φ holds	60
DoneWhen(δ, φ)	**self** has just done δ and φ was true when he started	60
AllPast(φ)	φ was true at all past times	60
SomePast(φ)	φ was true at some past time	60
AllFut(φ)	φ will be true at all future times	60
SomeFut(φ)	φ will be true at some future time	60
AllPastN(φ)	φ was true at all past times and is true **now**	60
and similarly for **SomePastN**, **AllFutN**, and **SomeFutN**		
Can(δ, φ)	**self** can achieve φ by doing δ	96