Handbook of Knowledge Representation
Edited by F. van Harmelen, V. Lifschitz and B. Porter
© 2008 Elsevier B.V. All rights reserved
DOI: 10.1016/S1574-6526(07)03015-5

Chapter 15

Reasoning about Knowledge and Belief

Yoram Moses

15.1 Introduction

An agent operating in a complex environment can benefit from adapting its behavior to the situation at hand. The agent's choice of actions at any point in time can, however, be based only on its local knowledge and beliefs. When many agents are present, the success of one's agent's actions will typically depend on the actions of the other agents. These, in turn, are based on the other agents' own knowledge and beliefs. It follows that to operate effectively in a setting containing other agents, an agent must, in addition to its knowledge about the physical features of the outside world, consider its knowledge about other agent's knowledge. This line of reasoning can be extended to justify the need for using deeper levels of knowledge, of course. Moreover, the task of obtaining relevant knowledge and that of affecting the knowledge of other agents, become important goals in many applications. This crucial connection between knowledge and action is what makes knowledge and belief two of the most frequently used notions in everyday discourse. It also suggests that rigorous frameworks for reasoning about knowledge and belief can be of value when analyzing scenarios involving multiple agents.

Philosophers have been concerned with *epistemology*, the study of knowledge, for thousands of years, going back to the great Chinese, Greek, and Indian thinkers. The focus of much of their analysis was on fundamental questions about the nature of knowledge: What can be known? When does someone know something? How does knowledge relate to truth and to belief? Rigorous logical treatment of knowledge and belief go back to the work of von Wright in the 1950's. It gained substantial grounding in Hintikka's seminal book *Knowledge and Belief* in 1962 [28], which based modal logics of knowledge on Kripke's possible-worlds semantic modeling of modal logics [30]. Hintikka's work was followed by a wave of research in the 1960's on logics for knowledge and belief and their proper axiomatizations, with a focus on the relationship between knowledge and belief [15, 31].

In the second half of the Twentieth century, researchers in different fields of science recognized the need to understand the role that knowledge and belief play in multi-agent systems and multi-agent interaction. In 1969, David K. Lewis published the book *Conventions*, which contained the first explicit definition of *common knowledge* among a set of agents (or individuals). Extensions of this work in the context of linguistics and the philosophy of language were made by Schiffer and by Clark and Marshall [9]. In game theory, Schelling [40] and Harsanyi [25–27] recognized the role that uncertainty plays in the analysis of games in the 1960's, and a model for knowledge and common knowledge in games was first presented by Aumann in 1976 [1]. In artificial intelligence, McCarthy argued for modeling agents' knowledge and beliefs as essential components of an agent in the mid-1970's. This ultimately gave rise to the BDI *Belief, Desire and Intentions* model of agent-hood [39] discussed in Chapter 24. The role of explicit, knowledge-based analyses of distributed protocols and distributed systems was initiated by Halpern and Moses in 1984 [17], while Goldwasser, Micali and Rackoff introduced tools for reasoning about cryptographical protocols that provide *zero knowledge* [16] in 1985. Since the 1980's, a large body of literature consisting of many books and hundreds if not thousands of conference and journal papers have been written continuing and extending these lines of research.

Surveying the state of the art is well beyond the scope of this chapter. Instead, this chapter aims at providing a somewhat biased introduction to the topic of reasoning about knowledge and belief, based in large part on the book *Reasoning about Knowledge* [12], where the reader is advised to seek further detail and additional references to the literature. Half of this chapter is devoted to introducing basic concepts, and the other half focuses on illustrating how the runs and systems framework can be set up to model multi-agent applications of interest. This involves properly matching the agents' behaviors or strategies, modeled by protocols, with a careful definition of the environment in which the agents operate, which is in turn modeled using the notion of a context.

15.2 The Possible Worlds Model

15.2.1 A Language for Knowledge and Belief

Before attempting to model knowledge and belief, we need to observe that these terms are used in many different senses in natural language. Thus, we may talk about knowing a language, knowing a profession, or knowing how to perform a particular task. We may *believe* in a higher power, or in a person. One may consider knowing what the time is, or knowing what sequence of numbers will open a safe. While all of these are perfectly reasonable uses of the terms in question and are worthy of investigation in their own right, we will focus on knowledge and belief in the truth of facts. Thus, we will be interested in expressing and modeling statements such as that "*agent i believes that it is midnight*", or that "*Alice knows that the key is hidden under the rug*". We will also be interested in statements such as "*Alice knows that Bob does not know that Alice knows that Bob spilled the beans*", of an agent's knowledge about other agents' knowledge, as well as issues having to do with what a group of agents knows.

Let Φ be a set of *primitive propositions*, standing for the basic facts that we wish to reason about in a given application of interest. The particulars of Φ typically depend on the application considered, and do not affect the general framework for reasoning about knowledge. We will therefore omit explicit mention of Φ if no confusion arises. Denote by $[n] = \{1, \ldots, n\}$ a set of agents. The language $\mathcal{L}_n^K = \mathcal{L}_n^K(\Phi)$ for knowledge among the agents in $[n]$ is defined to be the smallest set of formulas that contains Φ and is closed under the standard Boolean connectives $\wedge$ and $\neg$ (all other connectives of propositional logic can be expressed using $\wedge$ and $\neg$), and under modal operators K_i, for every $i \in [n]$.[1] Thus, every proposition $p \in \Phi$ is a formula of $\mathcal{L}_n^K$ and, inductively, if $i \in [n]$ while φ and ψ are formulas of $\mathcal{L}_n^K$, then $\neg\varphi$, $\varphi \wedge \psi$ and $K_i\varphi$ are formulas of $\mathcal{L}_n^K$. We also use standard abbreviations from propositional logic, such as $\varphi \vee \psi$ for $\neg(\neg\varphi \wedge \neg\psi)$, $\varphi \Rightarrow \psi$ for $\neg\varphi \vee \psi$, and $\varphi \Leftrightarrow \psi$ for $(\varphi \Rightarrow \psi) \wedge (\psi \Rightarrow \varphi)$. It is also convenient to use the notation **true** as shorthand for the tautologically true formula $p \vee \neg p$ and **false** as shorthand for $\neg$**true**. We read $K_i\varphi$ as "agent i knows φ". Thus, if p stands for "The lock on Bob's office door is broken", and we identify Alice with agent 1 and Bob with 2, then $K_1 p \wedge K_1\neg K_2 p$ will state that Alice knows that the lock is broken, and that she also knows that Bob does not know this. By further nesting of knowledge and Boolean operators it is possible to express more complex statements involving agents' knowledge about other agents knowledge (or lack thereof), etc. in $\mathcal{L}_n^K$. Indeed, these can quite quickly express statements that appear to be fairly tricky. Consider the formula $K_1K_2K_1p \wedge \neg K_2K_1\neg K_2K_1p$, stating that Alice knows that Bob knows that Alice knows p, and Bob does not know that Alice knows that Bob does not know that Alice knows p. Even such a short formula may require the listener to pause before its meaning is understood. We thus need a clear framework for interpreting such statements in a precise way.

Most rigorous approaches to modeling knowledge and belief capture these notions in terms of *possible-worlds semantics*. The idea here is that an agent in a given scenario is typically not omniscient regarding all aspects of the current state of the world. Rather, it considers many possibilities for the true state of the world. If, say, a given door is locked in all of the worlds that the agent considers possible, then the agent may be said to know (or believe) that the door is locked. More generally, agent i will know a fact φ if φ holds in all of the worlds that i considers possible. Conversely, φ is *not* known by i if i considers possible at least one world in which φ does not hold. Notice that knowledge is defined in terms of (a more primitive notion of) possibility. Clearly, the set of worlds an agent considers possible will generally be different in distinct states of the world. In particular, this set changes over time, as the state of the world changes, and as the agent learns new facts and perhaps forgets others. Following Hintikka, we model knowledge in terms of a Kripke structure $M = (S, \pi, \mathcal{K}_1, \ldots, \mathcal{K}_n)$, where S is a set of states of the world, $\pi : \Phi \to 2^S$ specifies for each primitive proposition the set of states at which the proposition holds, and $\mathcal{K}_i \subseteq S \times S$ is a binary relation on the states of the world where, intuitively, $(s, t) \in \mathcal{K}_i$ means that when the actual state of the world is s, agent i considers the world represented by t to be *possible*. Formulas of $\mathcal{L}_n^K$ are considered true or false at a world (M, s) consisting of a state s in a structure M. We denote by $(M, s) \models \varphi$ the fact that a formula φ is true, or *satisfied* at a world (M, s). The satisfaction relation $\models$ is formally defined by induction on the structure of φ.

[1] A similar language, $\mathcal{L}_n^B$, can be defined for reasoning about belief if we substitute the modal knowledge operators K_i by analogous belief operators B_i, for all $i \in [n]$. We will speak in terms of knowledge and make explicit mention of belief when this is warranted.

Primitive propositions $p \in \Phi$ form the base of the induction, and their truth is determined according to the assignment π:

$$(M, s) \models p \text{ (for a primitive proposition } p \in \Phi) \quad \text{iff} \quad s \in \pi(p).$$

Negations and conjunctions are handled in the standard way:

$$(M, s) \models \neg\psi \quad \text{iff} \quad (M, s) \not\models \psi.$$

$$(M, s) \models \psi \wedge \psi' \quad \text{iff} \quad \text{both } (M, s) \models \psi \text{ and } (M, s) \models \psi'.$$

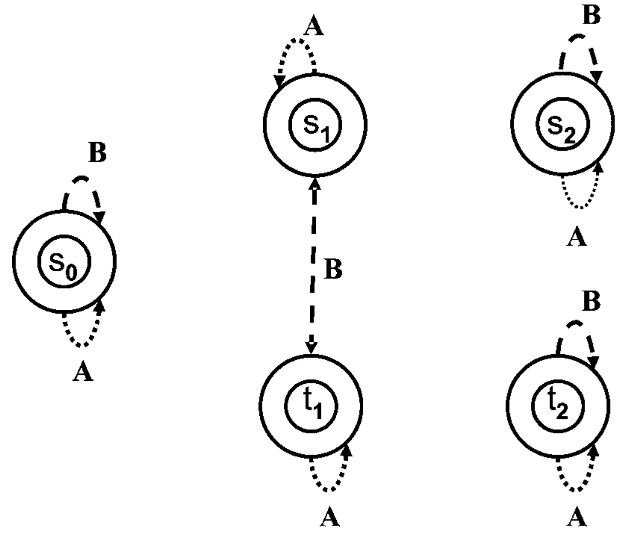
Finally, the crucial clause handles formulas of the form $\varphi = K_i\psi$. Here, the intuition that knowledge corresponds to truth in all possible worlds is captured by:

$$(M, s) \models K_i\psi \quad \text{iff} \quad (M, t) \models \psi \text{ for all } t \text{ such that } (s, t) \in \mathcal{K}_i.$$

A formula φ is said to be *valid in* (*the structure*) $M = (S, \ldots)$ if $(M, s) \models \varphi$ holds for all $s \in S$. Moreover, φ is called *valid* if it is valid in all structures M. We say that φ is *satisfiable* if $(M, s) \models \varphi$ holds for some M and s. It is not hard to verify that φ is satisfiable exactly if $\neg\varphi$ is not valid.

Observe that the set of $\mathcal{L}_n^K$ formulas that are true at a state s in a structure M depends on the $\mathcal{K}_i$ relations as well as on the assignment π. Two states can satisfy the same primitive propositions as determined by the assignment π, and yet differ considerably in the $\mathcal{L}_n^K$ formulas that they satisfy.

Example 15.1. To illustrate these definitions, let us consider a very simple example, involving two agents, named Alice and Bob. Initially, Alice has a coin and Bob is in the other room. Alice tosses the coin to the floor. (Nothing is known about the bias or fairness of the coin, except that it has two different faces.) Once Bob hears the coin hit the floor, he enters the room and observes whether the coin shows Heads or Tails. There are many ways to model this example using the possible-worlds framework we have discussed. We now present one particular choice. We model the scenario by way of a Kripke structure $M = (S, \pi, \mathcal{K}_A, \mathcal{K}_B)$. The set Φ of primitive propositions in M consists of three basic facts: $\Phi = \{\mathsf{Toss}, \mathsf{Heads}, \mathsf{Tails}\}$. Intuitively, Toss stands for the fact that the coin has been tossed, Heads holds if the coin toss resulted in the coin landing Heads, and Tails stands for the tossed coin having landed Tails. The model should allow us to reason about what is true (and what is known) at each of the three stages of this scenario: Initially, immediately after Alice tosses the coin, and finally after Bob enters the room. To this end, we consider the set of states $S = \{s_0, s_1, t_1, s_2, t_2\}$, where s_0 is the initial state, s_1 and t_1 are the intermediate states where the coin landed Heads and Tails, respectively, while s_2 and t_2 are the final states that occur following s_1 and t_1, respectively, once Bob has entered the room and he sees the tossed coin. The assignment π specifies what states the primitive propositions are true in, and is based on the above description. Consequently, $\pi(\mathsf{Toss}) = \{s_1, t_1, s_2, t_2\}$, $\pi(\mathsf{Heads}) = \{s_1, s_2\}$, and $\pi(\mathsf{Tails}) = \{t_1, t_2\}$. Finally, we need to define the possibility relations $\mathcal{K}_A$ and $\mathcal{K}_B$ over the states of S. Assuming that Alice can see the outcome of her coin toss immediately, and can see if Bob is in the room, in all states of this example she knows exactly what the actual state is. So $\mathcal{K}_A = \{(s, s) \mid s \in S\}$. Bob, in turn, knows the actual state at the first and third stages, and is unable to distinguish between the states at the second stage—after Alice tosses the coin but before he enters the room. Thus,



π(Heads) = $\{s_1,s_2\}$
π(Tails) = $\{t_1,t_2\}$
π(Toss) = $\{s_1,t_1,s_2,t_2\}$

Figure 15.1: Knowledge via possible worlds.

$\mathcal{K}_B = \{(s,s) \mid s \in S\} \cup \{(s_1,t_1),(t_1,s_1)\}$. A visual illustration of the structure M is given in Fig. 15.1, where the binary relations $\mathcal{K}_A$ and $\mathcal{K}_B$ are represented by directed edges labeled by A and B, respectively.

With this model for the coin-toss scenario, we can now establish the truth of some nontrivial statements in this model. One is

$$(M,s_0) \models K_A\neg\mathsf{Toss} \wedge K_B K_A K_B(\neg\mathsf{Heads} \wedge \neg\mathsf{Tails}),$$

capturing the fact that in the initial state s_0 Alice knows that the coin has not been tossed, and Bob knows that Alice knows that Bob knows the coin is currently showing neither Heads nor Tails; or

$$(M,s_1) \models \mathsf{Heads} \wedge \neg K_B\mathsf{Heads} \wedge \neg K_B\mathsf{Tails} \wedge K_B(K_A\mathsf{Heads} \vee K_A\mathsf{Tails}),$$

which establishes that at s_1 the coin is showing Heads, Bob does not know this, but Bob knows that Alice knows whether the coin shows Heads or Tails.

This example illustrates that explicitly constructing a model of knowledge even for a scenario with a simple structure and very little uncertainty may be quite laborious. In more interesting situations, the state space and the possibility relations quickly become much more complex. Notice that a number of choices and simplifying assumptions are built into modeling the scenario as we have done. One has to do with the granularity of the modeling. Intuitively, the states chosen for S in the above example "sample" the world at three distinct stages. Moreover, given the choice of primitive propositions in this example, the language is restricted to expressing facts about the coin tossing and its outcome (and knowledge about these). Thus, for example, while in all possible worlds in this model, both Bob and Alice know whether or not they are in the same room, we cannot express this without extending the set Φ of primitive propositions defined in the example. Finally, having a very small number of states in S automatically implies that agents have strong knowledge about each other's knowledge. We shall return to this issue later on once we define common knowledge.

Observe that both possibility relations $\mathcal{K}_A$ and $\mathcal{K}_B$ in the above example are equivalence relations: Reflexive, symmetric and transitive. This is not a coincidence. In many applications, it is natural to consider the knowledge of agent i at a state s as being based of some concrete *view* $v_i(s)$ that the agent is assumed to have at s. Two states s and t are then indistinguishable, so that $(s, t) \in \mathcal{K}_i$, exactly if $v_i(s) = v_i(t)$. The view $v_i(s)$ in such applications is typically a function of the agent's observations so far. It may consist, for example, of her complete history, what she sees in front of her, the state of her memory or the set of formulas in the agent's database. Possibility relations that are obtained in this fashion are automatically equivalence relations.

15.3 Properties of Knowledge

The possible-worlds approach to modeling knowledge and belief is quite popular and attractive, and view-based definitions of knowledge are natural in many applications. They turn out, however, to model the cognitive state of an idealized agent, as we shall see by analyzing the properties of knowledge and belief under these definitions. We capture the properties of knowledge by considering the valid formulas of $\mathcal{L}_n^K$.

Even before considering the definition of $\models$ for the knowledge operators, our definition inherits valid formulas from its propositional component, from the fact that the Boolean operators $\neg$ and $\wedge$ are treated as they are in propositional logic. We thus obtain that

A0. All instances of propositional tautologies are valid,

which we think of as the *Propositional Axiom*, and the inference rule of *Modus Ponens*:

MP. If φ is valid and $\varphi \Rightarrow \psi$ is valid, then ψ is valid.

Intuitively, the pair **A0** and **MP** ensure that our logic is an extension of propositional logic.

One central property that follows from the definition that $K_i\varphi$ holds if φ holds at all worlds that i considers possible is that an agent knows all the logical consequences of his knowledge. If an agent knows both φ and that φ implies ψ, then both φ and $\varphi \Rightarrow \psi$ are true at all worlds he considers possible. Thus ψ must be true at all worlds that the agent considers possible, so he must also know ψ. It follows that the *Distribution Axiom*

A1. $(K_i\varphi \wedge K_i(\varphi \Rightarrow \psi)) \Rightarrow K_i\psi$,

which states that knowledge is closed under implication, is valid. This is clearly a nontrivial assumption, which does not always match our intuitions regarding knowledge in everyday life.

Further evidence that our definition of knowledge assumes rather powerful agents comes from the fact that agents know all tautologies. In fact, they are guaranteed to know all formulas that are valid in the structure. If φ is true at all the worlds of a structure M, then φ must, in particular, be true at all the worlds that agent i considers possible at every world (M, s). Thus, $K_i\varphi$ must also hold at all worlds of M. More formally, we have the following *Knowledge Generalization Rule*:

G. For all structures M, if $M \models \varphi$ then $M \models K_i\varphi$.

While this implies that if φ is valid then so is $K_i\varphi$, this does *not* mean that the formula $\varphi \Rightarrow K_i\varphi$ is valid. The formula φ is valid in M only if it holds at all worlds in M. Indeed, it is quite common for a formula φ to hold, without $K_i\varphi$ being true. In the example above, at state s_1 the coin has landed Heads, but Bob does not know this. Notice that the Generalization rule can be applied repeatedly, and yield, for example (if repeated twice), that if $M \models \varphi$ then $M \models K_i K_j K_i\varphi$.

The Distribution Axiom **A1** and Generalization Rule **G** are forced by the possible-worlds modeling. They are shared by every normal modal operator [29]. It turns out that, in a precise sense, the logic K, consisting of axioms **A0** and **A1** and the rules **MP** and **G** completely characterizes the set of valid formulas of $\mathcal{L}_n^K$.

Considering the valid $\mathcal{L}_n^K$ formulas does not present "epistemic" properties for the K_i operators beyond those implied by the logic K. Thus, for example, there is no necessary connection between what is known and the facts that are true. This changes once we restrict the class of structures in a useful way. Recall from our discussion after Example 15.1 that if knowledge is derived in a view-based manner, then the possibility relations $\mathcal{K}_i$ are equivalence relations. We now turn to consider the set of formulas that are valid in this case. For the remainder of this section, we study validity with respect to the class of structures with equivalence possibility relations.

When possibility is an equivalence relation, each relation $\mathcal{K}_i$ is, in particular, reflexive. This means that, at every world (M, s), the current world is always one of the possible worlds. (In other worlds, since $(s, s) \in \mathcal{K}_i$, the world (M, s) is considered "possible" at (M, s).) From the definition of when $(M, s) \models K_i\varphi$ holds it follows that if an agent knows a fact, then it is true. More formally, the so-called *Knowledge Axiom*:

A2. $K_i\varphi \Rightarrow \varphi$

is valid. The Knowledge Axiom **A2** is often considered to be the central property distinguishing knowledge from belief. The intuition behind this is that while it possible to have false beliefs, known facts are necessarily true.

Two additional properties of knowledge that hold in this class of structures state that an agent has strong abilities to introspect into his own knowledge. An agent knows precisely which are the facts that he knows and which facts he does not know. These are captured by the *Positive Introspection Axiom* **A3** and the *Negative Introspection Axiom* **A4** given by:

A3. $K_i\varphi \Rightarrow K_i K_i\varphi$, and

A4. $\neg K_i\varphi \Rightarrow K_i\neg K_i\varphi$.

The Positive Introspection Axiom states that, if i knows φ then i knows that he knows φ, while the Negative Introspection Axiom states the converse: When i does not know φ, he knows that he does not know φ. Thus, while an agent may have only partial knowledge about what is true in the world, these axioms guarantee that he has perfect knowledge about his own knowledge. In the philosophy literature, **A3** is considered more acceptable as a property of knowledge and belief than **A4**. Both are determined to hold in structures in which the possibility relations are equivalence relations.

The collection of properties that we have considered so far—the Distribution Axiom, the Knowledge Axiom, the Positive and Negative Introspection Axioms, and the Knowledge Generalization Rule—has been studied in some depth in the literature. They are often called the S5 properties.

The axioms and rules discussed above are often viewed as an axiom system, with **MP** and **G** interpreted as *rules of inference*. Axiom systems provide a means for proving formulas. Given a set Γ of axioms, $\Gamma \vdash \varphi$ (or just "$\vdash \varphi$" if Γ is clear from context) stands for "φ is provable (from Γ)". In this context, **MP** is interpreted as saying that if $\vdash \varphi$ and $\vdash \varphi \Rightarrow \psi$ (so that both φ and $\varphi \Rightarrow \psi$ are provable), then we can conclude $\vdash \psi$ (so ψ can be considered provable). The rule **G** then states that from $\vdash \varphi$ we can conclude $\vdash K_i\varphi$.

The axiom system consisting of axioms **A0** and **A1** and the rules **MP** and **G** is called the modal logic K, and it is satisfied by any normal modal operator [29]. The full suite of axioms and rules above: **A0–A4** together with the rules **MP** and **G**, constitute the logical system S5, while removing **A4** yields the logical system S4. From the validity properties cited above, it is possible to show that every formula of $\mathcal{L}_n^K$ provable in K is valid in every Kripke structure $M = (S, \pi, \mathcal{K}_1, \dots, \mathcal{K}_n)$, while every formula provable in S5 is valid in every structure in which all of the possibility relations $\mathcal{K}_i$ are equivalence relations. That these axiom systems truly capture the set of valid formulas follows from the fact that the converse is also true: For either class of structures, every valid fact is provable from the corresponding axiom system.

In settings involving *belief*, rather than knowledge, the Knowledge Axiom **A2** is typically dropped, often replaced by the axiom:

A2′. $\neg K_i$ **false**.

If the possibility relations in a Kripke structure M are *serial*, meaning that for every $s \in S$ and $i \in [n]$ there exists a state $t \in S$ such that $(s, t) \in \mathcal{K}_i$, then axiom **A2′** is valid in M. If we replace **A2** by **A2′** in S5, we obtain the logic known as KD45.

15.4 The Knowledge of Groups

Formulas of $\mathcal{L}_n^K$ that contain several epistemic operators can often be thought of as describing states of knowledge of *groups* of agents. Thus, for example, $K_i p \wedge \neg K_j p$ describes a situation in which i and j have asymmetric knowledge about the truth of p. In the case of belief, the formula $B_i B_j p \wedge \neg B_j p$ describes a situation in which i has a misconception about j's beliefs. A state of knowledge that appears quite often in speech and in the analysis of multi-agent systems is captured by the $\mathcal{L}_n^K$ formula $K_1\varphi \wedge \cdots \wedge K_n\varphi$. This corresponds to *everyone* knowing φ. It is convenient to abbreviate this $\mathcal{L}_n^K$ formula by $E\varphi$, as this allows us to compactly write facts such as $E\varphi \wedge \neg EE\varphi$ in which "everyone knowing" is nested. It is not hard to see that $E\varphi$ and $EE\varphi$ are not equivalent. As a counterexample consider a formula ψ stating that a given team has won the world cup finals. If each of the agents learns of the outcome independently, say by hearing about it on the radio or reading a newspaper, then $E\psi$ clearly holds, but $EE\psi$ need not. More generally, let us define $E^1\varphi = E\varphi$ and inductively define $E^{k+1}\varphi = E(E^k\varphi)$ for $k \geqslant 1$. It can be shown that $E^{k+1}\varphi$ and $E^k\varphi$ are not, in general, equivalent. For every level k there is a world (M, s) and formula φ such that $(M, s) \models E^k\varphi \wedge \neg E^{k+1}\varphi$ (see [17]).[2]

15.4.1 Common Knowledge

While $E\varphi$ is expressible in $\mathcal{L}_n^K$, there are other natural states of group knowledge that are not. Perhaps the most important of these is *common knowledge*, which corresponds to everyone knowing a fact, everyone knowing that everyone knows it, etc. Let us denote by $C\varphi$ the fact that φ is common knowledge. Intuitively, we think of $C\varphi$ as satisfying

$$C\varphi \equiv E\varphi \wedge E^2\varphi \wedge \cdots \wedge E^k\varphi \wedge \cdots.$$

The right-hand side of this equivalence is an infinite conjunction. It is not an $\mathcal{L}_n^K$ formula, because all $\mathcal{L}_n^K$ formulas are finite. Fortunately, there are various ways to define common knowledge formally. Practically all of them coincide when interpreted using Kripke structures. One is the following. Given a Kripke structure $M = (S, \pi, \mathcal{K}_1, \ldots, \mathcal{K}_n)$, define let $\mathcal{E} = \bigcup_{i\in[n]} \mathcal{K}_i$. Thus, $\mathcal{E}$ is a binary relation over S consisting of every pair (s, t) such that $(s, t) \in \mathcal{K}_i$ for *some* i. It is easy to verify that

$$(M, s) \models E\psi \quad \text{iff} \quad (M, t) \models \psi \text{ for all } t \text{ such that } (s, t) \in \mathcal{E}.$$

It is straightforward to verify that E satisfies

$$E\varphi \equiv \bigwedge_{i\in[n]} K_i\varphi.$$

Even when all of the $\mathcal{K}_i$ possibility relations are equivalence relations, their union $\mathcal{E} = \bigcup_{i\in[n]} \mathcal{K}_i$ is not. The E operator will not, in general satisfy analogues of the Introspection Axioms **A3** and **A4**.

We now define the binary relation $\mathcal{C}$, which will correspond to common knowledge, to be the transitive closure of $\mathcal{E}$. Thus, $(s, t) \in \mathcal{C}$ if there is a sequence $s = s_0, s_1, \ldots, s_k = t$ such that $(s_i, s_{i+1}) \in \mathcal{E}$ holds for all $0 \leqslant i \leqslant k-1$. Notice that both $\mathcal{E}$ and $\mathcal{C}$ are completely determined by the $\mathcal{K}_i$ relations of a structure M. We extend $\mathcal{L}_n^K$ by closing off under the common knowledge operator C (so that in the inductive definition of formulas, if φ is a formula then so is $C\varphi$). Common knowledge is then formally defined by

$$(M, s) \models C\psi \quad \text{iff} \quad (M, t) \models \psi \text{ for all } t \text{ such that } (s, t) \in \mathcal{C}.$$

[2] It is often convenient to think of E as a state of knowledge of the group of all agents. Indeed, there are cases where a fact can be known to all members of a given set G of agents, but not to the rest. For example, we may be interested in whether all students of a given class know that the exam date has been changed. Within a larger framework, it might not be of interest to ensure that, say, everyone in the university knows this. When analyzing such scenarios, it is customary to use operators such as E_G, which stand for *everyone in G knows*. Similar restriction to groups of agents will be applicable to other states of knowledge that we discuss below.

This definition of common knowledge is essentially equivalent to the infinite conjunction of $E^k\varphi$ mentioned above. Indeed, if $(M,s) \models C\varphi$ holds then $(M,s) \models E^k\varphi$ holds for all $k \geqslant 1$. The converse is also true: If $(M,s) \models E^k\varphi$ holds for all $k \geqslant 1$, then $(M,s) \models C\varphi$. It is often convenient to think of common knowledge by viewing the Kripke structure as a graph (as is done in Fig. 15.1). The definition of $\mathcal{C}$ and $\mathcal{E}$ immediately implies that $(s,t) \in \mathcal{C}$ exactly if there is a directed path (possibly consisting of edges from $\mathcal{K}_i$'s of different agents) from s to t. In the special but commonly occurring case in which the $\mathcal{K}_i$'s are equivalence relations, $\mathcal{C}$ is also an equivalence relation, and its equivalence classes are precisely the connected components of the graph of M. A fact φ is then common knowledge at s exactly if it holds at all states in the connected component of s in the graph defined by M.

The notion of common knowledge appears to have been discussed informally in the sociology literature on the nature of consensus as early as 1967 [41]. It was given a formal definition, using the term *shared awareness*, by Friedell [13]. The name *common knowledge* was coined by the Philosopher by David K. Lewis in 1969 [32], who identified it as an inherent property of conventions. Later work in Game Theory [1], Linguistics [9] and Computer Science [17] showed the relevance of common knowledge to central issues in each of these fields. McCarthy suggested having a logic of knowledge in which common knowledge is represented by an agent he termed the *Fool* and common knowledge is then taken to be what "*any fool*" knows. Given our definition above, the possibility relation corresponding to the Fool's knowledge would be $\mathcal{C}$. We remark that if the $\mathcal{K}_i$ possibility relations in M are equivalence relations, then so is the relation $\mathcal{C}$ defined based on them. Thus, in settings in which knowledge satisfies the properties of $\mathsf{S5}$, the common knowledge operator satisfies $\mathsf{S5}$ as well. In particular, the Generalization Rule **G** now becomes

$$\text{If } M \models \varphi \text{ then } M \models C\varphi.$$

Let us return to our example concerning Alice and Bob and the precious coin, depicted in Fig. 15.1. Since the $\mathcal{K}_i$ relations in the example are equivalence relations, so is $\mathcal{C}$. The equivalence classes of $\mathcal{C}$ in this simple example are given by:

$$\mathcal{C} = \big\{\{s_0\}, \{s_1, t_1\}, \{s_2\}, \{t_2\}\big\}.$$

Thus, in the states s_0, s_2 and t_2, the actual state is the only possibility according to $\mathcal{C}$, and hence, in the corresponding worlds (M,s_0), (M,s_2) and (M,t_2), all true formulas are common knowledge. In both states of the $\mathcal{C}$-equivalence class $\{s_1,t_1\}$, Alice knows the outcome of the coin, and so $C(K_A\mathsf{Heads} \vee K_A\mathsf{Tails})$ holds at both. Similarly, since Bob does not know the outcome in either of the states, $C(\neg K_B\mathsf{Heads} \wedge \neg K_B\mathsf{Tails})$ holds at both states. Notice that the common knowledge about Alice knowing the outcome of the toss after the first stage in this example is built into the model. If the scenario were changed slightly to one in which Bob does not know that Alice has tossed the coin before he enters Alice's room in the second stage (states s_2 and t_2)—for example, suppose that Bob were to consider s_0 possible in both s_1 and t_1—then both agents would know the same about the propositional facts Toss, Heads and Tails at s_1 and t_1 as in the original example, but then there would be much weaker common knowledge regarding these facts at the two states. Since S is small in this example, and the connected components are even smaller, the amount of common knowledge here is considerable. Observe that in a given Kripke structure $M = (S, \ldots)$ only states in S can ever be reachable from (or in the same connected component as) a given state. Hence, it is common knowledge at all states of the model that no state outside of S is possible. E.g., if cars in all states of S cars are either Red or Green, it will be common knowledge that all cars are Red or Green (provided that Φ contains appropriate propositions that correspond to colors of cars). Moreover, Kripke structures with a small number of states typically model situations in which there is a great deal of common knowledge. It follows that modeling a situation in which agents have considerable uncertainty regarding each others' knowledge normally requires a fairly large set of states S to represent the agents' ignorance. In fact, even when there are only two agents and Φ consists of a single proposition, representing a sufficient degree of mutual ignorance (lack of knowledge about each other's knowledge) may require a Kripke structure of unbounded, even infinite, size.

Common knowledge is often thought to be such a strong state of knowledge, that people wonder whether it can be attained in practice. After all, achieving a small number of levels of knowledge about knowledge among more than one agent already seems quite complex. Intuitively, one might expect that to attain infinitely many levels of interactive knowledge would require an infinite interchange of messages, or a similar unreal feat. This intuition is, however, misguided. Under reasonable assumptions, common knowledge occurs quite frequently. A typical scenario in which common knowledge arises in a natural way is a setting in which some fact φ is "public", so that whenever φ holds, all agents know that it does. Such, for example, is the situation arising when two people shake hands. Inherent in the act of shaking hands is the fact that both parties know that they are shaking hands. In every world either agent considers possible, the handshake is taking place. It follows by induction that a handshake takes place in every world that is connected to the current one, and the agents thus have common knowledge of the handshake. A similar situation arises when someone makes a public announcement in a lecture hall, or when a couple shares a candlelight dinner. The intuition that "public" facts are common knowledge is formally captured by the *Induction Rule* for common knowledge, which is stated as:

Ind. If $M \models \varphi \Rightarrow E\varphi$ then $M \models \varphi \Rightarrow C\varphi$.

In practice, facts become common knowledge by becoming public in the sense of the induction rule. In many cases, in order to prove that ψ is common knowledge we find a stronger fact φ (so that $M \models \varphi \Rightarrow \psi$) to which the Induction Rule can be applied.

Another important property of common knowledge is captured by the *Fixedpoint Axiom* for common knowledge, which is in a way a converse of the Induction Rule, since it states that when a fact φ is common knowledge, then everyone knows φ and, moreover, everyone knows that φ is common knowledge:

CK. $C\varphi \Rightarrow E\varphi \wedge EC\varphi$.

The Fixedpoint Axiom captures an aspect of common knowledge that relates it to conventions and agreements: Whatever is common knowledge is automatically known by all to be common knowledge. At least at an intuitive level, this is a property we expect from conventions and agreements.

15.4.2 Distributed Knowledge

At the other end of the spectrum from common knowledge is *distributed knowledge*, which roughly corresponds to the knowledge that results from combining the knowledge of all agents, and considering them as one "super agent". If Alan knows the first six numbers in an eight-number sequence that won last week's lottery, and Beth knows the last three numbers in the sequence, then together Alan and Bet can be said to have distributed knowledge of the winning sequence. Despite the fact that neither Alan nor Beth knows the winning sequence by themselves.

We denote distributed knowledge by a modal operator D, and define

$$(M, s) \models D\psi \quad \text{iff} \quad (M, t) \models \psi \text{ for all } t \text{ such that } (s, t) \in \bigcap_i \mathcal{K}_i.$$

Intuitively, $\bigcap_i \mathcal{K}_i$ corresponds to combining the agents' knowledge because, for every state $s \in S$, each state that is known at s by at least one agent to be impossible, is also considered impossible according to the intersection. Thus, we can think of distributed knowledge as representing the knowledge that an agent with access to all agent's information would have.

The definition of satisfaction for Distributed knowledge has the same structure as that for K_i, but with respect to the possibility relation $\mathcal{D} = \bigcap_i \mathcal{K}_i$. It follows that D is a normal modal operator, so that it satisfies Axiom **A1** and the Generalization Rule **G**. Moreover, if all $\mathcal{K}_i$'s are equivalence relations, then their intersection is also an equivalence relation. In this case, distributed knowledge satisfies all of the properties of S5. An axiom connecting knowledge and distributed knowledge is:

Ad. $\models K_i\varphi \Rightarrow D\varphi$.

Using axioms **Ad** and **A1** we can show, for example, that $\models (K_i\varphi \wedge K_j\psi) \Rightarrow D(\varphi \wedge \psi)$. This is one property that we would expect the combined knowledge of the agents to satisfy. In particular, it can be used to establish that Alan and Beth know the winning sequence of lottery numbers in the example discussed above.

In actual applications, we are sometimes interested in states of knowledge of a subset of the agents. Thus, for example, if in our Alice and Bob example there was a third agent Chris that was in the second room with Bob and stayed there when Bob moved into Alice's room, then the outcome of the coin toss would be common knowledge to the subset consisting of Alice and Bob only. Depending on how we would modify the example in this case, the fact Toss that the coin has been tossed could be common knowledge among all three agents or just among Alice and Bob. In an analogous fashion, in particular, applications we may be interested in the knowledge distributed among a particular subset of the agents, and not only in the distributed knowledge for the set of all agents. To accommodate such finer distinctions concerning distributed knowledge and common knowledge, it is possible to subscript the C and D operators by a group $G \subseteq [n]$. The definitions for C_G and D_G are then modified by restricting attention to the possibility relations of the agents participating in G. The logical language obtained by closing $\mathcal{L}_n^K$ off with all operators C_G and D_G for common and distributed knowledge is denoted by $\mathcal{L}_n^{CD}$.

15.5 Runs and Systems

When reasoning about knowledge, one is often interested in modeling a dynamic situation, in which the world evolves, and with it the state of knowledge of the agents changes. In this section we consider a natural way to model these.

We think of every agent at any given instant as being in a well-defined *local state*. The precise structure and contents of this local state typically depends on the application. The local state in our setting captures all of the information that is available to the agent when it determines its next action. A *global state* corresponds to a snapshot of the state of the world frozen at an instant. Formally, it is modeled by an $(n+1)$-tuple of the form $(s_e, s_1, \dots, s_n)$, where s_i is i's local state, for $1 \leqslant i \leqslant n$. The additional state s_e is called the *local state of the environment*, and it accounts for all else that is relevant to the analysis, possibly keeping track of aspects of the world that are not part of any agent's local state. These may include, for example, messages in transit in a communication network, the state of entities that are not being modeled as agents in a given application (perhaps a traffic light), or even temporary properties of an agent that the agent might not be aware of. Intuitively, an agent's local state captures exactly what is visible to the agent at the current point. The agent is able to distinguish two points exactly if its local state in one is different from its local state in the other. Thus, we may think of a mailbox as belonging to (or even being part of) an agent in a given application. But if the agent accesses the contents of the mailbox only by performing an explicit *read* operation, then the contents of the mailbox at any given time are modeled as part of the environment state, and the local state may contain the result of actual reads the agent has performed. One final use of the environment state in many applications is for keeping track of various aspects of the history of the run. If different actions may lead to the same global states, or if an agent's state does not keep track of the actions the agent has performed, it is often convenient to add this information as part of the environment's state.

The evolution of a world over time produces a history, which in our terminology will be called a *run*. Formally, a run r is a function assigning every time instant t a global state $r(t)$. If $r(t) = (s_e, s_1, \dots, s_n)$ then we denote by $r_i(t)$ the local state s_i, for $i = e, 1, \dots, n$. It is often convenient to identify time with the natural numbers, in which case the run is identified with the sequence $r(0), r(1), \dots$. We typically reason about knowledge in a setting in which many different histories are possible, at least at the outset. The structure that represents these possibilities is called a *system*, and it is identified with a set R of possible runs. A possible world is now represented by a *point* (r, m) consisting of a run r at a time m. Viewed appropriately, a system induces a Kripke structure, and we can consider formulas as being true or false at a point (r, m) with respect to a system R. Given a set Φ of primitive propositions, we add an interpretation π that determines the truth of the propositions at every point in R. It is convenient to define π to be a function $\pi : \mathcal{G} \times \Phi \to \{\mathbf{true}, \mathbf{false}\}$, where $\mathcal{G}$ contains the global states in R. Once π is added, we can define the truth of all propositional formulas at points of a system in the standard way. We next define a notion of indistinguishability among points that induces possibility relations $\mathcal{K}_i$ for every agent i. We say that (r, m) and (r', m') are *indistinguishable* to agent i, denoted $(r, m) \sim_i (r', m')$, if $r_i(m) = r'_i(m')$. In words, an agent cannot distinguish among points exactly if it has the same local state in both. Observe that $\sim_i$ is an equivalence relation. A pair $\mathcal{I} = (R, \pi)$, which we call an *interpreted* system, now induces a Kripke structure $M_{\mathcal{I}}$ whose possibility relations are equivalence classes. Consequently, we can write $(\mathcal{I}, r, m) \models \varphi$ and say that $\varphi \in \mathcal{L}_n^K(\Phi)$ holds at (r, m) in (interpreted) system $\mathcal{I} = (R, \pi)$, if φ holds at (r, m) in the induced Kripke structure $M_{\mathcal{I}}$.

We now consider how Alice and Bob's coin-tossing example would be modeled as a system. The local states of each agent can be described by three observable parameters: (i) The current time (0, 1, 2), (ii) the room (ρ_1 or ρ_2) that the agent is in, and (iii) if the agent is in the first room ρ_1, and the coin has been tossed, then what face of the coin is showing. Alice would have five local states $a^0 = (0, \rho_1)$, $a^{1h} = (1, \rho_1, H)$, $a^{1t} = (1, \rho_1, T)$, $a^{2h} = (2, \rho_1, H)$ and $a^{2t} = (2, \rho_1, T)$, while Bob would have four local states $b^0 = (0, \rho_2)$, $b^1 = (1, \rho_2)$, $b^{2h} = (2, \rho_1, H)$, and $b^{2t} = (2, \rho_1, T)$. The environment state can be chosen in different ways. Indeed, in this particular example all of the relevant information is already captured in the local states of the agents; it is therefore possible to consider the environment state as being identically λ. In order to fit an extension of this modeling in Section 15.7.1 we will instead choose the environment's state to record the current time at each state. If we consider Alice as agent 1 and Bob as agent 2, then each of the states in Fig. 15.1 corresponds to a global state. Specifically, $s_0 = (a^0, b^0, 0)$, $s_1 = (a^{1h}, b^1, 1)$, $t_1 = (a^{1t}, b^1, 1)$, $s_2 = (a^{2h}, b^{2h}, 2)$ and $t_2 = (a^{2t}, b^{2t}, 2)$. The set of global states in the example is thus $\mathcal{G} = \{s_0, s_1, t_1, s_2, t_2\}$. The interpretation π for $\Phi = \{\mathsf{Toss}, \mathsf{Heads}, \mathsf{Tails}\}$ is the one defined in the original example and depicted in Fig. 15.1.

The system consists of two runs $R = \{r, r'\}$, where $r(0) = s_0$, $r(1) = s_1$, and $r(2) = s_2$, while $r'(0) = s_0$, $r'(1) = t_1$, and $r'(2) = t_2$. With respect to the interpreted system $\mathcal{I} = (R, \pi)$ the truth of epistemic formulas is now well-defined and works as expected. It is instructive to observe that the state s_0 which served to specify a world in the original Kripke structure is represented in $\mathcal{I}$ by two *different* points: $(r, 0)$ and $(r', 0)$. It is straightforward to verify that the exact same formulas of $\mathcal{L}_n^K(\Phi)$ hold at $(\mathcal{I}, r, 0)$ and $(\mathcal{I}, r', 0)$. As expected, these are the same formulas that are satisfied at (M, s_0) in the original example. What distinguishes these two points is the fact that they appear in different runs (histories). Indeed, the coin lands Heads in the future of $(r, 0)$ and lands Tails in that of $(r', 0)$. In the next section we enrich the language with temporal operators. Once we do this, the sets of formulas satisfied at the two points no longer coincide.

The runs and systems modeling of knowledge allows considerable control over the manner in which knowledge evolves over time. By varying the way in which events change the agents' local states, we can obtain different flavors of knowledge. Thus, for example, suppose that local states consist of a sequence of all events that the agent has observed so far. In this case, agents would have *perfect recall* and would not forget their past knowledge. Conversely, if information is removed from an agent's local state, then the agent can "forget" facts that it knows. This reflects the natural property that the evolution of knowledge depends on memory and how it is utilized. There are cases in which, during the design of a solution to a problem in a distributed setting, it is convenient to start out assuming that agents have perfect recall. An analysis in terms of knowledge is often simpler to perform in such a setting. Once a basic solution is obtained, it is typically possible to try to optimize the solution by reducing the size and contents of the local states, while maintaining the correctness of the solution. Such a scheme can be found in [7, 24, 37, 38, 35].

15.6 Adding Time

In many applications, it is natural to model time as ranging over the natural numbers (or a prefix of the natural numbers). In this case, a run is a sequence of global states. As shown in Chapter 12 linear-time temporal operators can thus readily be added to the language, and the satisfaction relation $\models$ can be defined for temporal operators in the standard way. For example, suppose that we add the operators O (standing for *at the next time instant*), $\Box$ (standing for *forever in the future*) and $\Diamond$ (standing for *eventually*) to the language. We denote the resulting language by $\mathcal{L}_n^{KT}$. Then we can define

$$(\mathcal{I}, r, m) \models \mathsf{O}\varphi \quad \text{iff} \quad (\mathcal{I}, r, m+1) \models \varphi,$$

and

$$(\mathcal{I}, r, m) \models \Box\varphi \quad \text{iff} \quad (\mathcal{I}, r, m') \models \varphi \quad \text{for all } m' \geqslant m.$$

The $\Diamond$ operator is treated as the dual of $\Box$, so that $\Diamond\varphi$ is considered as shorthand for $\neg\Box\neg\varphi$. Additional operators, such as Until and past operators can be added in a similar fashion.

In the Alice and Bob example we would now have, for example:

$$(\mathcal{I}, r, 0) \models \mathsf{O}\mathsf{Heads} \wedge K_B\mathsf{O}(\mathsf{Heads} \vee \mathsf{Tails}) \wedge K_B\Box(\mathsf{Tails} \Rightarrow K_A\mathsf{Tails}).$$

Once temporal operators are added to the logical language, we can express the fact that things will be known at times other than the present, and we can also express knowledge about temporal facts. Knowledge and time are complementary notions and, to a large extent, are orthogonal to each other. Indeed, temporal operators allow us to reason along the time axis within a run, while knowledge operators allow reasoning across runs (as well as, sometimes, within the run if local states repeat).

It is natural to seek an axiom system for the runs and systems model in terms of the temporal-epistemic language $\mathcal{L}_n^{KT}$. Clearly, knowledge satisfies S5, because it is determined based on possibility relations that are equivalence relations. Similarly, the temporal operators satisfy the axioms of standard linear-time temporal logic [34]. More interesting is the interaction between knowledge and time. For example, consider the formula

$$K_i\varphi \Rightarrow \mathsf{O}K_i\varphi, \tag{15.1}$$

which states that if agent i currently knows φ, then i will still know φ at the next state. This property can not be expected to hold for arbitrary formulas. For example, consider the proposition $\mathsf{time} = 3$ where $\pi(\mathsf{time} = 3) = \{(r, m) \colon m = 3\}$. If $K_i(\mathsf{time} = 3)$ holds at a given point, it would fail to hold one time step later, since $\mathsf{time} = 3$ would not hold at time 4. We say that a formula ψ is *stable* with respect to an interpreted system $\mathcal{I}$ if $\mathcal{I} \models \psi \Rightarrow \Box\psi$. Stable formulas are ones that, once true, are guaranteed to remain true forever. The argument showing that property (15.1) is not valid made use of the nonstable formula $\mathsf{time} = 3$. Is the property valid for stable formulas? Recall that an agent that does not have perfect recall may forget that it knew certain facts. It turns out that property (15.1) holds in systems in which the formula φ is stable, and agent i has perfect recall. We now make this claim more precise. Agent i's *local state sequence* at a point (r, m) is the sequence of local states obtained from $[r_i(0), r_i(1), \dots, r_i(m)]$ once we remove immediate repetition of states. Intuitively, if the agent's state does not change from one time instant to the next, then the agent cannot observe that time has passed. Consequently, according to agent i's subjective point of view, at two points in which the agent has the same local state sequence, it has had the same local history. We say that i has perfect recall in the system R if at all points of R, whenever $r_i(m) = r'_i(m')$ the agent i has the same local state sequence at (r, m) and at (r', m'). It is not hard to prove that $K_i\varphi \Rightarrow \bigcirc K_i\varphi$ is valid for stable formulas φ in systems in which i has perfect recall.

Let us consider another natural property relating knowledge and time:

$$K_i \bigcirc \varphi \Rightarrow \bigcirc K_i \varphi. \tag{15.2}$$

This formula states that if agent i knows that tomorrow φ will hold, then tomorrow the agent will know φ. This reasonable property is not a valid axiom for the runs and systems model, however. There are two factors that can render this formula false. One is the fact, discussed in the previous section, that agents might be forgetful. Thus, i may know something today, and no longer be aware tomorrow that this knowledge existed. (This can, for example, result from the agent deleting some messages from its mail file.) In particular, the agent can forget knowledge about what will be true at the next time instant. The second factor that can foil this property involves the agent's awareness of the passage of time. The "next" operator $\bigcirc$ refers to a point in time that is one time step into the future. There are systems in which agents are fully aware of the passage of time, and ones in which agents need not have perfect knowledge of it. A system is said to be *synchronous* if agents can always distinguish points at different times. In such a system, if $m \neq m'$ then $r_i(m) \neq r'_i(m')$ holds for all runs r and r' and agents i. For the class of synchronous systems, in which agents have perfect recall, the formula in (15.2) is indeed a valid axiom.

As these examples illustrate, the interaction between knowledge and time is subtle, and it depends on the particular assumptions one makes about properties of the system at hand. There has been extensive work on characterizing complete axiom systems for classes of interpreted systems with various sets of properties (see, e.g., [23]).

15.6.1 Common Knowledge and Time

The fixedpoint axiom for common knowledge implies that $\models C\varphi \Rightarrow EC\varphi$, which intuitively means that common knowledge is inherently "public": Everyone knows what is common knowledge. Since knowledge satisfies the Knowledge Axiom, at any given moment either nobody knows $C\varphi$ or all agents do. An agent cannot come to know $C\varphi$ before $C\varphi$ holds, and all agents do. Thus, the transition from $\neg C\varphi$ to $C\varphi$ requires a *simultaneous* change in the local states of all (relevant) agents. It follows that in systems where it is not possible to coordinate simultaneous transitions (or at least to identify a simultaneous transition once it has occurred), it is impossible for facts that are not commonly known to become common knowledge. In particular, no common knowledge can arise in asynchronous systems or when communication is not reliable [17, 12, 5].

This raises a philosophical issue that has practical modeling implications. Recall that we typically think of common knowledge as arising naturally from public or shared events such as a handshake. But can we really say that the agents come to know that they are shaking hands simultaneously? Apparently not. Indeed, it may very well be the case that the tactile sensation of the handshake reaches one agent's brain two milliseconds before it reaches the second agent's brain. And even if the sensations arrive truly at the same instant, the agents cannot reasonably rule out the possibility that they arrived at slightly different times. In fact, if time is modeled at a sufficiently fine granularity then real systems do not allow for simultaneity, and hence also not for common knowledge. When we choose the model for a given problem, however, it often makes sense to keep the model as simple as possible to faithfully model the situation (but no simpler). In such a model, we may well find common knowledge arising. Thus, for example, in the synchronous models mentioned earlier, where $r_i(m) = r'_i(m')$ can hold only if $m = m'$, we immediately obtain that the current time is always common knowledge, provided Φ is expressive enough to talk about the current time (e.g., has propositions of the form $\mathsf{time} = m$ for $m = 0, 1, \ldots$). Moreover, in synchronous systems in which messages are guaranteed to take exactly k time steps to be delivered, when Alice receives a message from Bob, they share common knowledge that she has received this message, provided that Bob remembers the message and its sending time for at least k time units. The common knowledge paradox comes from the fact that as the model becomes more detailed, transitions are no longer simultaneous, and common knowledge vanishes [17, 12].

15.7 Knowledge-based Behaviors

15.7.1 Contexts and Protocols

As we have seen, each systems and runs model directly induces a Kripke structure and consequently allows reasoning about knowledge and belief. But where do the systems come from? In many applications, we wish to reason about knowledge in a given setting in which the agents are following particular strategies, or programs. The system corresponding to such a scenario consists of all possible runs (histories) that can arise. Strategies, or programs, are not executed in a void. Rather, they are carried out within a particular context. This context determines how Nature, or the environment, evolves and interacts with the behavior of the agents. More formally, suppose that we fix sets of local states $L_e, L_1, \ldots, L_n$ and local actions $\mathsf{ACT}_e, \mathsf{ACT}_1, \ldots, \mathsf{ACT}_n$ for the agents and the environment. A *protocol* for $i = e, 1, \ldots, n$ is a function $P_i : L_i \rightarrow (2^{\mathsf{ACT}_i} \setminus \{\{\,\}\})$ determining, possibly in a nondeterministic fashion, what action i performs as a function of its local state. If $P_i(\ell_i) = A$, then A is a nonempty set A of actions, and the action performed by i when in state ℓ_i will be one of the members of A. If $P_i(\ell_i)$ is a singleton for all $\ell_i \in L_i$, then P_i reduces to a function from states to actions, and is thus a deterministic protocol.

Let $\mathcal{G} = L_e \times L_1 \times \cdots \times L_n$. In order to reason about knowledge and belief, we typically assume a fixed set Φ of primitive propositions, and an interpretation $\pi : \mathcal{G} \times \Phi \rightarrow \{\mathbf{true}, \mathbf{false}\}$. A *joint action* is a tuple $\vec{\mathsf{a}} = (a_e, a_1, \ldots, a_n)$ consisting of an action $a_i \in \mathsf{ACT}_i$ for $i = e, 1, \ldots, n$. We will assume that at every point each of the agents performs an action. The fact that in many applications we do not think of all agents as moving at all times can be handled by assuming that the environment actions can influence the scheduling of which agent actions are enabled and hence may in fact affect the global state.

A *context* is a tuple $\gamma = (\mathcal{G}_0, \tau, P_e)$, where $\mathcal{G}_0 \subset \mathcal{G}$ is a set of *initial* global states, τ is a *transition function*, mapping every global state g and joint action $\vec{\mathsf{a}}$ to a global state g'. Intuitively, if $\tau(g, \vec{\mathsf{a}}) = g'$ then the result of $\vec{\mathsf{a}}$ being performed in g is that the global state becomes g'. Finally, P_e is a protocol (often nondeterministic) for the environment. In many applications in which the environment's protocol P_e is nondeterministic, it is often natural to assume that the context also ensures certain *fairness* of the actions performed by the environment over time. We shall soon discuss how a fourth component can be added to the context to handle such cases.

We now turn to consider how protocols for the agents give rise to runs and systems. Define a *joint protocol* to be a tuple $\vec{P} = (P_1, \ldots, P_n)$ associating a protocol with each one of the agents, but not with the environment. We say that a run r is a *run of $\vec{P}$ in the context γ* If $r(0) \in \mathcal{G}_0$, and at every point (r, m) there is a joint action $\vec{\mathsf{a}}$ consisting of local actions $a_i \in P_i(r_i(m))$, for $i = e, 1, \ldots, n$, such that $\tau(r(m), \vec{\mathsf{a}}) = r(m + 1)$. Intuitively, this means that the runs begins in a legal state according to γ, and it proceeds at every step in a legal fashion: there is a joint action that can be generated by P_e and $\vec{P}$ that can transform the global state into the successor, according to the transition function τ. We can thus define the system $R(\vec{P}, \gamma)$ generated by $\vec{P}$ in γ to consist of the set of all runs of $\vec{P}$ in γ. Moreover, assuming that we have a fixed Φ and interpretation π in mind for $\mathcal{G}$, the corresponding interpreted system is $\mathcal{I}(\vec{P}, \gamma) = (R(\vec{P}, \gamma), \pi)$. Observe that in our framework contexts and joint protocols play complementary roles. Taken together, they give rise to a single, well-defined, interpreted system. This framework allows us to consider running a given protocol in different contexts, and similarly allows us to compare different protocols being run in the same system.

Let us briefly outline how Alice and Bob's coin tossing example can be represented in the framework that we have just described. The local states and global states are as described in Section 15.5, and the set of initial states is $\mathcal{G}_0 = \{s_0\}$. Let $\mathsf{ACT}_e = \{\mathsf{skip}, \mathsf{land\_heads}, \mathsf{land\_tails}\}$, $\mathsf{ACT}_1 = \{\mathsf{skip}, \mathsf{flip}\}$, and $\mathsf{ACT}_2 = \{\mathsf{skip}, \mathsf{move}\}$. The transition function is such that skip is the null action for all three entities, move changes Bob's location from room ρ_2 to ρ_1, flip is a toss of the coin by Alice, and land_heads and land_tails are environment actions that determine what side the coin will land on, in case the coin is flipped. To complete the description of the context γ we need to determine the environment's protocol P_e. We take it to perform skip at times 1 and 2, while at time 0 it prescribes a nondeterministic choice from the set $\{\mathsf{land\_heads}, \mathsf{land\_tails}\}$. Notice that the current time is a component in the local states of the environment, as well as of those of Alice and Bob. Hence, protocols defined as a function of the time (or round number) are in particular functions of the local states, as required. The joint protocol $\vec{P} = (P_1, P_2)$, where Alice's protocol P_1 performs flip at time 0 and skip at times 1 and 2. Finally, Bob's protocol P_2 performs skip at times 0 and 2, while performing move at time 1. It is straightforward to check that the interpreted system $\mathcal{I}$ from Section 15.5 is precisely $\mathcal{I} = \mathcal{I}(\vec{P}, \gamma)$ for the protocol and context just described.

We mentioned above the occasional need to add a fourth component to the context. An example is a distributed computer system in which communication is reliable, but there is no bound on message delivery times. Thus, every message that is sent is guaranteed to reach its destination. The environment's protocol at any given instant may nondeterministically choose between delivering a message that is in transit or not delivering it yet. But our assumption about the context is that the message may not remain in transit until the end of time. When such issues need to be accounted for, we add a fourth component Ψ to the context, where Ψ is an *admissibility condition* specifying the set of "acceptable" runs. Now r is a run of $\vec{P}$ in $\gamma = (\mathcal{G}_0, \tau, P_e, \Psi)$ if r is a run of $\vec{P}$ in the larger context $\hat{\gamma} = (\mathcal{G}_0, \tau, P_e)$, **and** r satisfies Ψ. Thus, runs that do not comply with Ψ do not arise in γ.

The admissibility condition Ψ in a context $\gamma = (\mathcal{G}_0, \tau, P_e, \Psi)$ should be *non-exclusive*, which means that for every protocol $\vec{P}$ for the agents, every finite prefix of a run of $\vec{P}$ in $\hat{\gamma} = (\mathcal{G}_0, \tau, P_e)$ must be a prefix of a run of $\vec{P}$ in $\gamma = (\mathcal{G}_0, \tau, P_e, \Psi)$. Roughly speaking, this ensures that Ψ captures aspects of the environment's infinite behavior, and does not influence the possible finite executions of protocols in the context.

15.7.2 Knowledge-based Programs

The close connection between knowledge and action is a central motivation for reasoning about knowledge. In fact, there are many settings in which it is natural to think of particular choices of actions or strategies as being triggered by an agent's knowledge. Thus, for example, an agent called Noah may start to build and ark if he knows that a flood is soon to come; his neighbors, who may not be privy to such knowledge, may instead prefer to herd their sheep and dismiss the looming danger. The essential role that knowledge plays in determining the actions that agents perform often makes knowledge into a goal in its own right: Indeed, a central goal of communication among agents typically has to do with ensuring that particular agents obtain certain knowledge (or beliefs). Conversely, many personal, financial, and political activities are required to satisfy secrecy constraints, which in turn mean that certain agents do not obtain particular knowledge. In some of these cases, it is useful to reason about actions at the knowledge level. Suppose that Bob does not know that Alice knows where to meet him for Dinner. His goal is then be simply to ensure that she comes to know where to meet him for Dinner. Alice's decision on where to go for Dinner may depend on whether she knows where Bob reserved a table. In this particular example, it may not matter to Bob whether he informs Alice by phone, by way of a messenger, or indeed by other means. The mode of communication that he uses is merely one way of implementing his goal, which is best thought of at the knowledge level. Assuming that Bob will not rest until he is confident that Alice has obtained this piece of knowledge, Alice may also wish to notify Bob once she has received his message. Again, the essential property Alice would wish to ensure is conveniently stated as a formula of $\mathcal{L}_2^K$—at the knowledge level. As this brief example illustrates, parts of everyday activity involves planning and acting to achieve goals that are best expressed at the knowledge level. One convenient tool for reasoning at the knowledge level is provided by *knowledge-*

based programs (kb programs, for short). We can think of a kb program for an agent i as having the form

```
if $\kappa_1$ then $\mathsf{a}_1$,
if $\kappa_2$ then $\mathsf{a}_2$,
...
if $\kappa_m$ then $\mathsf{a}_m$,
```

where each knowledge test κ_j is a Boolean combination of formulas of the form $K_i\varphi$, where $\varphi \in \mathcal{L}_n^K$ may contain nested occurrences of K_j operators. We assume that the tests $\kappa_1, \kappa_2, \dots, \kappa_m$ are mutually exclusive and exhaustive, so that exactly one will evaluate to true whenever i performs an action. Denote this particular program by Pg_i. Suppose that Alice and Bob typically dine at the restaurant R_2, but they had previously agreed that on this particular day Bob would try to make a reservation at *Chez Panis* (restaurant R_1). Once Bob manages to reserve a table, his program (which he performs repeatedly) may be:

```
if $\neg K_B K_A \mathsf{Reserved\_at}(R_1)$ then phone Alice and tell her
```

The program that Alice follows at 8 pm may then be

```
if $K_A \mathsf{Reserved\_at}(R_1)$ then go to $R_1$,
if $\neg K_A \mathsf{Reserved\_at}(R_1)$ then go to $R_2$.
```

Knowledge-based programs are very similar in form to standard computer programs. The main difference is that actions in kb programs are determined based on the actor's knowledge, rather than on the values of her local variables or computer memory. If we fix an interpreted system $\mathcal{I}$ for evaluating the truth of knowledge tests, a kb program such as Pg_i induces a unique deterministic protocol for i. Indeed, for every knowledge test κ_j in Pg_i, there is a set $L_i^{\kappa_j}(\mathcal{I})$ for i such that $(\mathcal{I}, r, m) \models \kappa_j$ exactly if $r_i(m) \in L_i^{\kappa_j}(\mathcal{I})$. Denoting agent i's current local state by ℓ_i, once $\mathcal{I}$ is fixed, the kb program reduces to the following standard program $\mathsf{Pg}_i^{\mathcal{I}}$:

```
if $\ell_i \in L_i^{\kappa_1}(\mathcal{I})$ then $\mathsf{a}_1$,
if $\ell_i \in L_i^{\kappa_2}(\mathcal{I})$ then $\mathsf{a}_2$,
...
if $\ell_i \in L_i^{\kappa_m}(\mathcal{I})$ then $\mathsf{a}_m$.
```

We identify $\mathsf{Pg}^{\mathcal{I}}$ with the protocol it induces. The question of what we should choose as the system $\mathcal{I}$ here is somewhat delicate. As we have seen, it is natural to think of a protocol or program as generating a well-defined system in a given context γ. This system is the set of runs of the program. In the case of a kb program, however, we need the system in order to figure out what the program that generates the system is! We can avoid this vicious circularity by considering a kb program as a *specification* which may or may not be implemented by a given standard program. Fix a context γ. A given joint protocol $\vec{P}$ generates a unique interpreted system $\mathcal{I}^{\vec{P}} = \mathcal{I}(\vec{P}, \gamma)$ in this context. We can now say that $\vec{P}$ *implements* $\mathsf{Pg} = (\mathsf{Pg}_1, \dots, \mathsf{Pg}_n)$ in (γ) if $\vec{P}$ is equivalent to the program $\mathsf{Pg}^{\mathcal{I}^{\vec{P}}}$. Intuitively, if $\vec{P}$ implements Pg, then we are justified in viewing all agents in $\vec{P}$ as acting according to, or following, the knowledge-based program Pg.

Knowledge-based programs are indeed specifications, in the sense that some have a unique implementation, some have many different implementations, and some will have no implementation in a given context. We now consider an example in which a natural knowledge-based program has two different implementations.

15.7.3 A Subtle kb Program

Consider a mobile robot controlling a rail cart that travels on a track with discrete locations numbered $0, 1, 2, \ldots$. The cart starts out at location 0 and can move only in the positive direction. The cart's motion is determined by the environment, and the robot can only control whether to stop the cart. Moreover, the robot has no memory, and it has access only to an imperfect location sensor. For every location $q \geqslant 0$, it is guaranteed that whenever the robot is at location q, the sensor-reading σ will be one of $\{q-1, q, q+1\}$. The robot's goal is to stop the cart at one of the locations 4, 5, or 6. Stopping outside this region is not allowed. What should the robot's program be? Define a proposition goal that is true at all points at which the robot's location is $q \in \{4, 5, 6\}$. Intuitively, as long as the robot does not know that goal holds, it should not halt. On the other hand, once the robot does know that goal holds, it should be able to stop the cart in the goal region, as desired. Thus, $K_r\mathsf{goal}$ seems to be both a necessary and a sufficient condition for stopping in the goal region.

More formally, we assume that the environment's local state consists of the current location (q), while the robot's local state consists solely of the sensor reading (σ) (recall that the robot cannot recall the past—it is assumed to be memoryless). The environment actions are $\mathsf{ACT}_e = \{\mathsf{stay}, \mathsf{move}\} \times \{-1, 0, 1\}$, where the first component determines whether or not the robot's cart will be moved one position to the right on the track, and the second component determines how the sensor reading σ is related to the actual position q. The robot's actions are $\mathsf{ACT}_r = \{\mathsf{skip}, \mathsf{halt}\}$, where the action halt overrules the environment's action, the cart stops at the current location, and never moves again. The environment's protocol P_e is as follows. At all times m not of the form $m = k^{100}$, the protocol prescribes a nondeterministic choice among the actions of ACT_e. At the few times m of the form $m = k^{100}$, the choice is restricted to $\{\mathsf{move}\} \times \{-1, 0, 1\}$—so that, if not halted, the robot moves one step to the right. It is straightforward to define the transition function τ that matches this description, thereby completing the definition of the context γ_r. We take $\Phi = \{\mathsf{goal}\}$, and π assigns goal the value true at $r(m) = (q, \sigma)$ exactly if $q \in \{4, 5, 6\}$.

Consider the knowledge-based program Rob

if $K_r(\mathsf{goal})$ **then** halt.

Clearly, Rob guarantees that the robot will never halt the cart outside of the goal region. But does it also guarantee that the robot always succeeds in halting in the goal region? The answer is not clear cut.

The properties of the sensor in this context ensure that, for every protocol P executed in γ_r we have $\mathcal{I}(P, \gamma_r) \models (\sigma = 5) \Rightarrow K_r(\mathsf{goal})$. This suggests that the following standard program, denoted Rob_s,

if $\sigma = 5$ **then** halt,

should be an implementation of Rob in γ_r. It is not hard to check that this is indeed the case. Unfortunately, this implementation of Rob does not guarantee that the robot halts in the goal region. There are many runs of Rob_s in this context in which $\sigma \neq 5$ holds throughout the run, despite the fact that the robot crosses the goal region and exits it. It follows that, in spite of its "obviousness", the knowledge-based program Rob does not guarantee that the robot will succeed in γ_r. It may appear that this situation is unavoidable. However, there is a twist to the story. For consider now the program Rob'_s:

if $\sigma > 4$ **then** halt.

Following this program, the robot will never stop before reaching position $q = 4$, because $\sigma > 4$ is not satisfied when $q < 4$. Moreover, when following this program, the robot *is* guaranteed to stop the cart if it ever reaches the position $q = 6$, since at that point the sensor reading must satisfy $\sigma \in \{5, 6, 7\}$, so that the condition $\sigma > 4$ is true. Finally, the environment's protocol in γ_r guarantees that, if the robot has not halted the cart, then the cart will be at position 6 no later than time $6^{100} + 1$.

The protocol described by Rob'_s is a standard implementation of the kb-program Rob. Thus, Rob has two qualitatively different implementations, with one being guaranteed to reach the goal in every run, while the other is not. This justifies considering knowledge-based programs as specifications that can be satisfied in different ways. We remark that small changes in the assumptions of this example can change the outcome of the analysis. In particular, if we change γ_r so that the robot has perfect recall, then Rob_s is *no longer* an implementation of Rob, and the protocol described by Rob'_s is the only implementation, and a good one at that.

Admittedly, our assumption about the environment's protocol being forced to perform at least k move actions in every k^{100} steps was somewhat unnatural. It was intended to capture the idea that if the robot does not perform a halt action, then it will eventually move beyond any finite point. A somewhat cleaner, alternative way, to capture this would have been to add an admissibility condition Ψ to the context, which would admit only runs for which the following temporal formula holds at the initial state: $\varphi = (\Diamond\mathsf{halt} \vee \Box\Diamond\mathsf{move})$: A run is admissible if either the robot eventually halts, or it is moved infinitely often.

In summary, the robot example shows a fairly natural scenario in which a knowledge-based program can have more than one implementation. As we have mentioned, there are kb programs that have no implementations, ones that have a single implementation, and ones that have many implementations. Fortunately, there are many cases in which a knowledge-based program is guaranteed to have a unique implementation. This happens, for example, in a context in which the agents have a global clock and the knowledge tests in the program do not refer to the future (see [11]). Indeed in our robot example, if the robot's local state contains the round number in addition to the sensor reading, then the only implementation of the knowledge-based program Rob is the more efficient Rob'_s.

The definition of implementation for knowledge-based programs that we have presented is fairly strict, because the agent following such a program must be able to evaluate all knowledge tests at all times. This is good for certain types of analyses and may be overkill in certain applications. There are also variations on knowledge-based programs [8] that use a more liberal notion of implementation, in which the knowledge tests are replaced by *sound* standard tests which, when true, guarantee that the tested knowledge actually exist. A standard test "implementing" a knowledge test in this case is allowed to fail when the agent does have the tested knowledge.

15.8 Beyond Square One

So far we have discussed the basic possible-worlds setting, considered the basic properties of knowledge and belief, and considered how the runs and systems (protocols & contexts) framework can be used to capture the knowledge and belief aspects of an application. There are hundreds of contributions to the literature that deal with the analysis of properties of knowledge and belief, and the properties of their logics. Other contributions apply reasoning about knowledge to particular domains such as distributed computing systems, multi-agent planning, philosophical puzzles, or game theory. There are various approaches and formalisms for modeling knowledge, some similar to our description, and others quite distinct from it. For example, in game theory the accepted model for knowledge is influenced by the terminology of probability theory, with the possibility relations typically being defined by associating with each agent (or player) a partition on the states of the universe [1]. Cells of the partition are equivalence classes, and the outcome is essentially an instance of the familiar S5-knowledge.

A more recent formalism for reasoning about knowledge is based on marrying possible-worlds modeling for knowledge with Dynamic Logic [3, 2, 4]. Here the idea is to explicitly model the effect that action have on the state of knowledge of the agents. Thus, for example, a public announcement of φ by a trustworthy agent causes φ to become common knowledge; hence, the state immediately following such an announcement satisfies $C\varphi$.

The properties of knowledge as captured by the S5 axioms are not considered an acceptable characterization of human knowledge. Clearly, the Propositional Axiom **A0** which states that all tautologies are known to all agents assumes an idealized notion of agent. Perhaps equally objectionable is the property captured by the Distribution Axiom **A1**, which states that an agent's knowledge is essentially closed under logical deduction. These properties are generally termed *logical omniscience*. They are unreasonable not only for describing the knowledge of humans, but also when an agent's knowledge is meant to be accessible by some form of tractable computation. We remark that the success obtained by using the formalism introduced in this chapter so far in treating a variety of problems in different domains was possible mainly in cases where logical omniscience was not the main issue to contend with. For example, in a setting where Alice receives an acknowledgment from Bob that he has received a particular message sent to him, there is no conceptual problem with using the formal conclusion that she *knows* he has received the message.

There have been many attempts at defining weaker variants of knowledge that will not suffer from logical omniscience. Some of these are syntactic, in which what is known may be a list of formulas [22]. Others involve a syntactic element of *awareness*, which gives rise to a distinction between implicit and explicit knowledge. Traditional (possible-worlds based) knowledge is thought of as being implicit, and an agent that implicitly knows φ and is also *aware* of φ, is considered as having explicit knowledge of φ [10]. In other approaches, the limitations on knowledge are either based on explicit resource bounds of the agents [36, 19], or are based on agents having access to an explicit set of algorithms for computing knowledge [20]. In the latter case, $K_i\varphi$ would hold at a given point if applying one of the algorithms at its disposal at that point can establish that it does.

15.9 How to Reason about Knowledge and Belief

We have defined logics for the language $\mathcal{L}_n^K$ of knowledge, and discussed axioms for common knowledge, distributed knowledge, and time. All of these are modal logics, and one might hope to be able to use general-purpose methods to reason about them. In fact, however, there are many hurdles to doing so. First of all, even for the logics we have considered for the basic language $\mathcal{L}_n^K$, deciding the satisfiability of formulas is PSPACE-complete [18]. The complexity of logics of knowledge and time depends on our assumptions about perfect recall, synchrony, and the properties of communication. In [21] Halpern and Vardi consider ninety six logics. In all cases the complexity of the satisfiability problem ranges between the intractable exponential time and the undecidable Π_1^1. Second, the properties of knowledge and its interaction with related modal operators such as time are very sensitive to the features of the system in question, or to those of the underlying context. They can differ significantly from one application to another. We have seen how issues such as whether communication is synchronous or asynchronous, and whether agents have perfect recall can affect the axioms. Other structural properties of a given system can make a significant difference. For example, in a given application φ might be local to agent i—so that $\varphi \equiv K_i\varphi$ is valid in the system. In another, Eve might receive a copy of every message exchanged between Alice and Bob. This imposes specific but sometimes crucial structure on the way knowledge can evolve in the system. Because of the richness of systems and contexts, no single set of axioms completely characterizes the properties of knowledge is a wide variety of applications. Recall that in the standard runs and systems framework of Sections 15.5 and beyond knowledge satisfies the axioms of S5. Since in any given system additional properties may hold, it follows that S5 provides properties that are *sound* in all such systems, but it does not in general completely characterize knowledge in the system. It follows that decision procedures for modal logics of, say, knowledge and time are not likely to be helpful for reasoning about multi-agent systems in practice.

Much closer to the nature of reasoning we are interested in is the notion of *model-checking* [6], which has proven very successful for reasoning about temporal properties of finite-state systems and is widely used in the hardware industry. As shown in [22], the model-checking problem of model checking an $\mathcal{L}_n^K$ formula φ with m symbols at a point of a structure with K points is bounded above by mnK^2. This appears much more tractable than deciding satisfiability, but since K might be large, this could still be a considerable challenge. In fact, as model-checking techniques and optimizations improve, there is hope for variations on this theme. One of the best approaches for reasoning about knowledge in multi-agent systems appears to be the use of special-purpose model-checkers designed for the task such as [14]. Another approach that is gaining in popularity has to do with adapting existing theorem provers or model checkers such as PVS, SMV or MOCHA to the epistemic domain [33]. In most cases the classes of systems that can be treated is limited in some way—in the number of states involved, or in the diversity of actions that can be applied. The MCK tool has the unique feature that while the underlying context being modeled is finite state, the agents' local states can grow unbounded. The field of tools and case studies is growing rapidly and will most likely yield practical results in the coming years.

15.9.1 Concluding Remark

This chapter presented some of the basic notions having to do with knowledge and belief in multi-agent systems. Its main focus and use to the reader may be as a short introduction to the task of modeling knowledge and belief in systems. There is a huge body of work in the area that we did not have time to even hint at. We believe that knowledge-based analyses of multi-agent systems, and reasoning about knowledge and belief will find more and more applications in the coming years and decades, and will continue to develop rapidly. For additional material beyond the cited references, the reader may also consult [42, 43, 45–49].

Bibliography

[1] R.J. Aumann. Agreeing to disagree. *Annals of Statistics*, 4(6):1236–1239, 1976.

[2] A. Baltag and L.S. Moss. Logics for epistemic programs. *Synthese*, 139(2):165–224, 2004.

[3] J. van Benthem. Epistemic logic: from knowledge to cognition. In Y. Moses, editor. *Proceedings 4th Conf. on Theoretical Aspects of Reasoning about Knowledge*, pages 167–168. Morgan Kaufmann Publishers, 1992.

[4] J. van Benthem, J. van Eijck, and B. Kooi. Logics of communication and change. *Information and Computation*, 204(11):1620–1662, 2005.

[5] K.M. Chandy and J. Misra. How processes learn. *Distrib. Comput.*, 1(1):40–52, 1986.

[6] E.M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. The MIT Press, 1980.

[7] C. Dwork and Y. Moses. Knowledge and common knowledge in a Byzantine environment: crash failures. *Information and Computation*, 88(2):156–186, 1990.

[8] K. Engelhardt, R. van der Meyden, and Y. Moses. Knowledge and the logic of local propositions. In I. Gilboa, editor. *Proceedings 7th. Conf. on Theoretical Aspects of Rationality and Knowledge*, pages 29–41. Morgan Kaufmann Publishers, 1998.

[9] H.H. Clark and C.R. Marshall. Definite reference and mutual knowledge. In A.K. Joshi, B.L. Webber, and I.A. Sag, editors. *Elements of Discourse Understanding*, pages 10–63. Cambridge University Press, Cambridge, 1981.

[10] R. Fagin and J.Y. Halpern. Belief, awareness and limited reasoning. *Artificial Intelligence*, 34(1):39–76, 1987.

[11] R. Fagin, J.Y. Halpern, Y. Moses, and M.Y. Vardi. Knowledge-based programs. *Distributed Computing*, 10(4):199–225, 1997.

[12] R. Fagin, J.Y. Halpern, Y. Moses, and M.Y. Vardi. *Reasoning about Knowledge*. The MIT Press, 2003.

[13] M.F. Friedell. On the structure of shared awareness. *Behavioral Science*, 14(1):28–39, 1969.

[14] P. Gammie and R. van der Meyden. MCK: Model checking the logic of knowledge. In *Proc. Computer Aided Verification, 16th International Conference (CAV)*, pages 479–483, 2004.

[15] E. Gettier. Is justified true belief knowledge? *Analysis*, 23(6):121–123, 1963.

[16] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[17] J.Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *J. ACM*, 37(3):549–587, 1990. An early version in *Proc. 3rd ACM Conf. on Principles of Distributed Computing*, 1984, pages 50–61.

[18] J.Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54(3):319–379, 1992.

[19] J.Y. Halpern, Y. Moses, and M.R. Tuttle. A knowledge-based analysis of zero knowledge. In *Proceedings 12th ACM Symp. on Theory of Comput. (STOC)*, pages 132–147, 1988.

[20] J.Y. Halpern, Y. Moses, and M.Y. Vardi. Algorithmic knowledge. In R. Fagin, editor. *Proceedings 5th. Conf. on Theoretical Aspects of Reasoning about Knowledge*, pages 255–266. Morgan Kaufmann Publishers, 1994.

[21] J.Y. Halpern and M.Y. Vardi. The complexity of reasoning about knowledge and time, I: Lower bounds. *Journal of Computer Systems Science*, 38(1):195–237, 1991.

[22] J.Y. Halpern and M.Y. Vardi. Model checking vs. theorem proving: A manifesto. In J. Allen, R.E. Fikes, and E. Sandewall, editors, *Proc. 2nd Int. Conf. on Principles of Knowledge Representation and Reasoning, KR'91*, pages 325–334, 1991.

[23] J.Y. Halpern, M.Y. Vardi, and R. van der Meyden. Complete axiomatizations for reasoning about knowledge and time. *SIAM Journal on Computing*, 33(3):674–703, 2004.

[24] J.Y. Halpern and L.D. Zuck. A little knowledge goes a long way: Knowledge-based derivations and correctness proofs for a family of protocols. *Journal of the ACM*, 39(3):449–478, 1992.

[25] J. Harsanyi. Games of incomplete information played by Bayesian players I. *Management Science*, 14:159–182, 1967.

[26] J. Harsanyi. Games of incomplete information played by Bayesian players II. *Management Science*, 14:320–334, 1967.

[27] J. Harsanyi. Games of incomplete information played by Bayesian players III. *Management Science*, 14:486–502, 1968.

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

[29] G.E. Hughes and M.J. Cresswell. *A New Introduction to Modal Logic*. Routledge, 1996.

[30] S.A. Kripke. A completeness theorem in modal logic. *J. Symbolic Logic*, 24:1–14, 1959.

[31] W. Lenzen. Recent work in epistemic logic. *Acta Philosophica Fennica*, 30, 1978.

[32] D. Lewis. *Convention: A Philosophical Study*. Harvard University Press, Cambridge, 1969.

[33] A. Lomuscio and F. Raimondi. MCMAS: A model checker for multi-agent systems. In *Proc. 12th TACAS*, pages 450–454, 2006.

[34] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, 1992.

[35] T. Mizrahi and Y. Moses. Continuous consensus via common knowledge. In *Proceedings 10th Conf. on Theoretical Aspects of Rationality and Knowledge*, pages 236–252, 2005. *Distributed Computing*, 2008, submitted for publication.

[36] Y. Moses. Resource-bounded knowledge. In M.Y. Vardi, editor. *Proceedings 2nd Conf. on Theoretical Aspects of Reasoning about Knowledge*, pages 261–275. Morgan Kaufmann Publishers, 1988.

[37] Y. Moses and M.R. Tuttle. Programming simultaneous actions using common knowledge. *Algorithmica*, 3(1):121–169, 1988.

[38] G. Neiger and M.R. Tuttle. Common knowledge and consistent simultaneous coordination. *Distributed Computing*, 6(3):181–192, 1993.

[39] S.J. Rosenschein and L.P. Kaelbling. The synthesis of digital machines with provable epistemic properties. In *Proc. 1st Conference on Theoretical Aspects of Reasoning about Knowledge*, pages 83–98. Morgan Kaufmann Publishers, 1986.

[40] T. Schelling. *The Strategy of Conflict*. Harvard University Press, 1960.

[41] T.J. Scheff. Towards a sociological theory of consensus. *American Sociological Review*, 37:32–46, 1967.

Further reading

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

[43] R. Demolombe and M.P. Pozos Parra. A simple and tractable extension of situation calculus to epistemic logic. In *Proceedings of the Twelfth International Symposium on Methodologies for Intelligent Systems (ISMIS-2000)*, pages 515–524, 2000.

[44] S.A. Kripke. Semantical analysis of modal logic I: Normal modal propositional calculi. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 9:67–96, 1963.

[45] J.-J.Ch. Meyer and W. van der Hoek. *Epistemic Logic for AI and Computer Science*. Cambridge University Press, 1995.

[46] R.C. Moore. A formal theory of knowledge and action. In J.R. Hobbs and R.C. Moore, editors. *Formal Theories of the Commonsense World*, pages 319–358. Ablex, Norwood, NJ, 1985.

[47] R. Parikh and R. Ramanujam. Distributed processes and the logic of knowledge. In *Logic of Programs*, pages 256–268, 1985.

[48] R. Petrick and H. Levesque. Knowledge equivalence in combined action theories. In *Proc. of KR-02*, 2002.

[49] R.B. Scherl and H.J. Levesque. Knowledge, action, and the frame problem. *Artificial Intelligence*, 144, 2003.