Knowledge, Probability, and Adversaries
(Preliminary Report)

Joseph Y. Halpern
IBM Almaden Research Center
San Jose, CA 95120
halpern@ibm.com

Mark R. Tuttle
MIT Laboratory for Computer Science
Cambridge, MA 02139
tuttle@lcs.mit.edu

Abstract: What should it mean for an agent to know or believe an assertion is true with probability .99? Different papers [FH88, FZ88, HMT88] give different answers, choosing to use quite different probability spaces when computing the probability an agent assigns to an event. We show that each choice can be understood in terms of a betting game, and that each choice corresponds to betting against a different opponent. We consider three types of adversaries. The first selects the outcome of all nondeterministic choices in the system; the second represents the knowledge of the agent's opponent (this is the key place the papers mentioned above differ); the third is needed in asynchronous systems to choose the time the bet is placed. We illustrate the need for considering all three types of adversaries with a number of examples. Given a class of adversaries, we show how to assign probability spaces to agents in a way most appropriate for that class, where "most appropriate" is made precise in terms this betting game. We conclude by showing how different assignments of probability spaces (corresponding to different opponents) yield different levels of guarantees in coordinated attack.

The second author was supported by an IBM Graduate Fellowship, and in part by the National Science Foundation under Grant CCR-86-11442, by the Office of Naval Research under Contract N00014-85-K-0168, and by the Defense Advanced Research Projects Agency (DARPA) under Contract N00014-83-K-0125.

1 Introduction

Probabilistic systems consist of a collection of agents interacting in the presence of some source of randomness (such as a fair coin). The precise meaning of an agent, of course, depends on the system under consideration: an agent may be a processor in a distributed system or a consumer in an economic model. Cryptographic and other probabilistic protocols make various guarantees to these agents about the probabilities of various events occurring. A number of recent papers have tried to formalize reasoning about knowledge and probability in such systems. The intent, in part, is to provide a framework within which these guarantees can be described and analyzed. Fagin and Halpern [FH88] present an abstract model for knowledge and probability in which they assign to each agent and each state a probability space to be used when computing the probability, according to that agent at that state, that a formula φ is true. They do not tell us how to *choose* these probability spaces, although they do show that more than one choice may be reasonable. One particular (and quite natural) choice is made in [FZ88] and some arguments are presented for its appropriateness; another is made in [HMT88] and used to analyze interactive proof systems.

In this paper we attempt to clarify this issue. We argue that no single probability space is appropriate in all contexts; different probability spaces can be viewed as most appropriate in the context of betting with different adversaries. From this point of view, a statement such as "I know event E will happen with probability at least α" is meaningless until the adversary is specified. Knowing that E holds with probability .99 against a weak adversary is not as good as knowing that E holds with probability .99 against a stronger adversary.

Adversaries actually play three fundamentally different roles in our framework. We briefly describe

these three roles here, and explore them in greater depth in the rest of the paper.

Typically when we analyze probabilistic protocols, we do so in terms of probabilities on the *runs* or *executions* of the protocol. When we say a protocol is correct with probability .99, we mean the protocol will do the right thing in .99 of the runs. A closer analysis of the situation reveals some subtleties. In fact, we do *not* have a probability distribution on the entire set of runs. For example, in an algorithm like Rabin's primality testing algorithm [Rab80], we typically do not assume a distribution on the inputs (the numbers to be tested). The only source of probability comes from the coin tosses. This means that for every fixed input, there is a probability space on the runs of the protocol on that input, rather than there being one probability space on the set of all runs. We can view the choice of input as a nondeterministic choice to which we do not assign a probability. Thus, we prove the algorithm works correctly with high probability for each initial nondeterministic choice. A similar situation arises in probabilistic protocols that are designed to work in the presence of a nondeterministic (perhaps adversarial) scheduler (e.g., [Rab82]). Again, we do not wish to assume some probability of playing a given scheduler. Instead, we factor out the choice of scheduler and prove that the protocol is correct with high probability for each scheduler.

This, then, is the first role played by the adversary: to factor out the nondeterminism in the system, allowing us to place a well-defined probability on the set of runs for each fixed adversary. We remark that this need to factor out the nondeterminism is implicit in most analyses of probabilistic protocols, and appears explicitly in [Rab82, Var85, FZ88].

The probability on the runs can be viewed as giving us an *a priori* probability of an event, before the protocol is run. However, the probability an agent places on runs will in general change over time, as a function of information received by the agent in the course of the execution of the protocol. New subtleties arise in analyzing this probability.

Consider a situation with three agents p_1, p_2, and p_3. Agent p_2 tosses a fair coin at time 1 and observes the outcome at time 2, but agents p_1 and p_3 never learn the outcome. What is the probability according to p_1 that the coin lands heads? Clearly at time 1 it should be 1/2. What about at time 2? There is one argument that says the answer should be 1/2. After all, agent p_1 does not learn any more about the coin as a result having tossed it, so why should its probability change? Another argument says that after the coin has been tossed, it does not make sense to say that the probability of heads is 1/2. The coin has either landed heads or it hasn't, so the probability of the coin landing heads is either 0 or 1 (although agent p_1 does not know which). This point of view appears in a number of papers in the philosophical literature (for example, [VF80, Lew80]). Interestingly, this issue even has implications for quantum mechanics (see [Mer85]).

We claim that these two choices of probability are best explained in terms of an adversary. This point comes out clearly if we consider betting games. If an agent believes an event E has probability α, then it should be willing to accept a payoff of $\$1/\alpha$ for a bet of \$1 on E (assuming it is risk neutral). Certainly at time 1, agent p_1 should be willing to accept an offer from either p_2 or p_3 to bet \$1 for a payoff of \$2 if the coin lands heads. On the other hand, agent 1 is clearly not willing to accept such an offer from p_2 at time 2 (since p_2 would presumably offer the bet only when it is sure it will win), although it is still willing to accept this bet from p_3. The point here is that in a betting game, not only is your knowledge important, but also the knowledge of the opponent offering the bet. Betting games are not played in isolation!

Thus, the second role played by the adversary in our framework is to model the knowledge of the opponent offering a bet to an agent at a given point in the run. One obvious choice is to assume you are playing against someone whose knowledge is identical to your own. This is what decision theorists implicitly do when talking about an agent's *posterior* probabilities [BG54]; it is also how we can understand the choice of probability space made in [FZ88]. By way of contrast, the choice in [HMT88] corresponds to playing someone who has complete knowledge about the past and knows the outcome of the coin flip; this corresponds to the viewpoint that says that when the coin has landed, the probability of heads is either 0 or 1 (although you may not know which).

A further complication arises when analyzing asynchronous systems. In this case there is a precise sense in which the agent does not even know exactly when the event to which it would like to assign a probability is being tested. Thus we need to consider a third type of adversary in asynchronous systems, whose role is to choose the time. We give an example of an asynchronous system where there are a number of plausible answers to the question "What is the probability the most recent coin toss landed heads?". It turns out that the different answers again correspond to playing different classes of adversaries. We remark that the case of asynchronous systems is also considered in [FZ88]. We can understand the assignment of "confidence" made there as corresponding to playing against a certain class of adversaries of this

third type.

Having shown that different definitions of probabilistic knowledge correspond to different classes of adversaries, we show, given a class of adversaries, how to construct a definition most appropriate for this class. We formalize our intuition concerning the probability an agent assigns to an event in terms of a betting game between the agent and an adversary. We show that our "most appropriate" definition has the property that it enables an agent to break even in this game, and any other definition with this property must correspond to an adversary even *more* powerful than the actual adversary. These results form the technical core of our paper.

The rest of the paper is organized as follows. In the next section, Section 2, we provide a formal model of a distributed system. In Section 3 we consider the problem of putting a probability on the runs of a system; this is where we need the first type of adversary, to factor out the nondeterministic choices. In Section 4 we start to consider the issue of how probability should change over time. In Section 5 we consider the choices that must be made in a general definition of probabilistic knowledge. In Section 6 we consider particular choices of probability assignments that seem reasonable in synchronous systems. Here we consider the second type of adversary, representing the knowledge of the other party in the betting game. In the full paper, we consider asynchronous systems, where we also here we have to consider the third type of adversary. In Section 7 we apply our ideas to analyzing the coordinated attack problem, showing how different notions of probability correspond to different levels of guarantees in coordinated attack.

2 Modeling systems

In order to study probability in a distributed system, we need a formal definition of such a system. Our model is that of [HF88], a simplification of [HM84]. We briefly describe it here.

Consider an arbitrary system of n interacting agents $p_1, \ldots, p_n$. Intuitively, a run of a system is a complete description of one of the possible interactions of the agents. Such an interaction is uniquely determined by the sequence of global states through which the system passes as a result of the interaction. Formally, a *global state* is an $(n+1)$-tuple $(s_e, s_1, \ldots, s_n)$ of local states, where s_e is the local state of the environment, and s_i is the local state of agent p_i. Loosely speaking, the environment component of the global state is intended to capture everything relevant to the state of the system that cannot be deduced from the agents' local states. A *run* of the system is mapping r from time to global states. We assume for sake of convenience that times are nonnegative integers. We identify a *system* with the set $\mathcal{R}$ of all possible runs of the system, intuitively the set of all possible interactions of the system agents. We denote the global state at time k in run r by $r(k)$, the local state of p_i in $r(k)$ by $r_i(k)$, and the local state of the environment by $r_e(k)$. We refer to the ordered pair (r,k) consisting of a run r and a time k as a *point*.

A fact is considered to be true or false of a point. We identify a fact φ with the set of points at which φ is true, and write $(r,k) \models \varphi$ iff φ is true at (r,k).[1] In a system $\mathcal{R}$, a fact φ is said to be a *fact about the run* if, given two points of the same run, φ is either true at both points or false at both points. Similarly, a fact φ is said to be a *fact about the global state* if, given two points with the same global state, φ is true at both points or false at both points.

We now define what it means for an agent to know a fact φ at a point (r,k) of a system $\mathcal{R}$. Intuitively, $r_i(k)$ captures all of agent p_i's information at (r,k). We say p_i considers a point (r',k') *possible* at (r,k), and write $(r,k) \sim_i (r',k')$, if p_i has the same local state at both points; that is, if $r_i(k) = r'_i(k')$. Following [HM84] (and many other papers since then), we say p_i *knows* φ at (r,k) if φ is true at all points p_i considers possible at (r,k). This means p_i knows φ at (r,k) if φ is guaranteed to hold given the information recorded in p_i's local state at (r,k). More formally, we denote the fact that p_i knows φ at (r,k) by $(r,k) \models K_i\varphi$, and define $(r,k) \models K_i\varphi$ iff $(r',k') \models \varphi$ for all $(r',k') \in \mathcal{K}_i(r,k)$, where $\mathcal{K}_i(r,k)$ is the set of all points in $\mathcal{R}$ agent p_i considers possible at (r,k). While this definition of knowledge depends heavily on the system $\mathcal{R}$ (it restricts the set of points an agent considers possible at a given point), the system will always be clear from context and we omit explicit reference to $\mathcal{R}$ in our notation.

3 Probability on runs

In order to discuss the probability of events in a distributed system, we must specify a probability space. In this section we show that in order to place a reasonable probability distribution on the runs of a system, it is necessary to postulate the existence of a class of *adversaries*.

[1] In Section 5 we define a logical language for describing such facts. Formally, a fact is the interpretation of a formula in such a language. See [HM84] for a complete formal treatment of the syntax and semantics of such a language.

Consider the simple system consisting of a single agent who flips a fair coin once and halts. This system consists of two runs, one in which the coin comes up heads and one in which the coin comes up tails. There is an obvious probability distribution on these two runs induced by the coin toss: each is assigned probability 1/2.

Now consider the system (suggested by Moshe Vardi; a variant also appears in [FZ88]) consisting of two agents, p_1 and p_2, where p_1 has an input bit and two coins, one fair coin landing heads with probability 1/2 and one biased coin landing heads with probability 2/3. If the input bit is 0, p_1 flips the fair coin once and halts. If the input bit is 1, p_1 flips the biased coin and halts. This system consists of four runs of the form (b, c) in which the input bit has value b and the outcome of the coin toss is c. What is the appropriate probability distribution on the runs of this system? For example, what is the probability of heads?

Clearly the probability of heads is 1/2 if the input bit is 0 and 2/3 if the input bit is 1, but we cannot make sense of the probability of heads until we are told how the input bit is set. In order to avoid thinking about nonprobabilistic events in a probabilistic system, it is conceivable one would be willing to assume the existence of a fixed distribution on the input bit, say one guaranteeing each input is equally likely. With this assumption, we can compute that the probability of the input bit being 0 is $\frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{2}{3} = \frac{7}{12}$. Often, however, such an assumption leads to results about a system that are simply too weak to be of any use. Knowing an algorithm produces the correct answer in .99 of its runs when all inputs are equally likely is of no use when the algorithm is used in the context of a different distribution on the inputs (one can imagine, for example, a primality-testing algorithm used in the context of different cryptographic protocols).

To overcome this problem, one might be willing to assume the existence of some fixed but unknown distribution on the inputs. Proving that an algorithm produces the correct answer in .99 of the runs in the context of an unknown distribution, however, is no easier than proving that for each fixed input the algorithm is correct in .99 of the runs, since it is always possible for the unknown distribution to place all the probability on the input for which the algorithm performs particularly poorly. Here the advantage of viewing the system as a single probability space is lost, since this is precisely the proof technique one would use when no distribution is assumed in the first place. In fact, unless we are willing to assume a distribution on the distributions themselves, we are simply moving the problem of dealing with a nondeterministic choice of inputs up one level to dealing with the problem of a nondeterministic choice of distributions!

This discussion leads us to conclude that some choices in a distributed system must be viewed as inherently nondeterministic (or, perhaps better, nonprobabilistic), and that it is inappropriate, both philosophically and pragmatically, to model probabilistically what is inherently nondeterministic. But then how can we reason probabilistically about a system involving both nondeterministic and probabilistic choices? Our solution—which is essentially a formalization of the standard approach taken in the literature—is to factor out initial nondeterministic events, and view the system as a collection of subsystems, each with its natural probability distribution. In the coin flipping example above, we would consider two probability spaces, once corresponding to the input bit being 0 and the other corresponding to the input bit being 1. The probability of heads is 1/2 in the first space and 2/3 in the second.

We want to stress that although this example may seem artificial, analogous examples frequently arise in the literature. In a probabilistic primality testing algorithm, for example, we do not want to assume a probability distribution on the inputs. We want to know that for all choices of input, the algorithm gives the right answer with high probability. In this situation, the natural thing to do is to partition the runs of the algorithm into a collection of subsystems, one for each possible input, and prove the algorithm gives the right answer with high probability in each of these subsystems (and, indeed, this is precisely what is done in both [Rab80, SS77]).

In many contexts of interest, the choice of input is not the only source of nondeterminism in the system. Later nondeterministic choices may also be made throughout a run. In asynchronous distributed systems, for example, it is common to view the choice of the next processor to take a step or the next message to be delivered as a nondeterministic choice. Similar arguments to those made above can be used to show that we need to factor out these nondeterministic choices in order to use the probabilistic choices (coin flips) to place a well-defined probability on the set of runs. A common technique for factoring out these nondeterministic choices is to assume the existence of a scheduler deterministically choosing (as a function of the history of the system up to that point) the next processor to take a step (cf. [Rab82, Var85]). It is standard practice to fix some class of schedulers, perhaps the class of "fair" schedulers or "polynomial-time" schedulers, and argue that for every scheduler

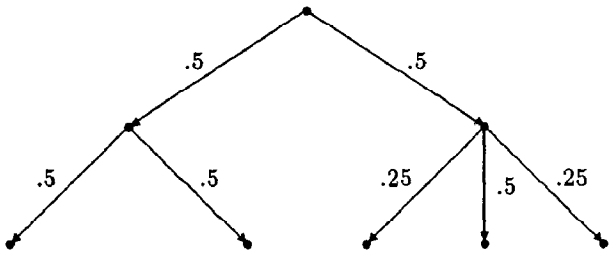

Figure 1: A computation tree.

in this class the system satisfies some condition.

If we view all nondeterministic choices as under the control of some adversary from some class of adversaries, then there is a straightforward way to view the set of runs of a system as a collection of probability spaces, one for each adversary. By fixing an adversary we factor out the nondeterministic choices and are left with a purely probabilistic system, with the obvious distribution on the runs determined by the probabilistic choices made during the runs. This is essentially the approach taken in [FZ88].

Once we fix the adversary we can represent the set of runs of the system as a set of *computation trees*, one associated with each adversary (see Figure 1). Nodes of the tree are global states, and edges of the tree are labeled with positive real numbers such that for every node the values labeling the node's outgoing edges sum to 1. (We assume every node has at most a countable number of outgoing edges.) Intuitively, the value labeling an outgoing edge of node s represents the probability the system makes the corresponding transition from node s.[2] The runs of the system are the paths in the tree. We can view each tree as a probability space in a natural way, since all nondeterministic choices have been factored out. Given a finite path in the tree, the probability of the set of runs extending this finite path is simply the product of the probabilities labeling the edges in this finite path.

Given an adversary A, we denote the computation tree corresponding to A by $\mathcal{T}_A$. Notice that $\mathcal{T}_A$ may be viewed as a probability space, a tuple $(\mathcal{R}_A, \mathcal{X}_A, \mu_A)$ where $\mathcal{R}_A$ is the set of runs in $\mathcal{T}_A$, $\mathcal{X}_A$ consists of subsets of $\mathcal{R}_A$ that are *measurable* (that is, the ones to which a probability can be assigned; these are generated by starting with sets of runs with a common finite prefix and closing off under countable union and complementation), and a probability function μ_A defined on sets in $\mathcal{X}_A$ so that the probability of a set of runs with a common prefix is the product of the probabilities labeling the edges of the prefix. If we restrict attention to finite runs (as is done in [FZ88]), then it is easy to see that each individual run is measurable, so that $\mathcal{X}_A$ consists of all possible subsets of $\mathcal{R}_A$. Moreover, the probability of a run is just the product of the transition probabilities along the edges of the run.

Formally, we define a *probabilistic system* to consist of a collection of computation trees (which we view as separate probability spaces); that is, a collection of probability spaces of the form $\mathcal{T}_A = (\mathcal{R}_A, \mathcal{X}_A, \mu_A)$, where A ranges over some set $\mathcal{A}$ of adversaries. We assume that the environment component in each global state in $\mathcal{T}_A$ encodes the adversary A and the entire past history of the run. This technical assumption ensures that different nodes in the same computation tree have different global states, and that we cannot have the same global state in two different computation trees. Given a point c, we denote the computation tree containing c by $\mathcal{T}(c)$. Our technical assumption guarantees that $\mathcal{T}(c)$ is well-defined.

The choice of the appropriate set of adversaries $\mathcal{A}$ against which the system runs is typically made by the system designer when specifying the correctness conditions the system is to satisfy. An adversary might be limited to choosing the initial input of the agents (in which case the set of possible adversaries would correspond to the set of possible inputs) as is the case in the context of primality-testing algorithms in which an agent receives a single number (the number to be tested) as input. On the other hand, an adversary may also determine the order in which agents are allowed to take steps, the order in which messages arrive, or which processors fail. One might also wish to restrict the computational power of the adversary to polynomial time. It depends on the application.

4 Probability at a point

We are interested in understanding knowledge and probability in distributed systems. An agent's knowledge varies over time, as its state changes. We would expect the probability an agent assigns to an event to vary over time as well. Clearly an agent's probability distribution at a given point must somehow be related to the distribution on runs if it is to be

[2] We represent only transitions that have positive probability. Thus, we are assuming here that there is a discrete probability on the set of possible transitions at each node. This means, for example, that we are disallowing the possibility that the next step will be a random assignment to a variable x chosen with uniform probability from the interval $[0,1]$. We could easily extend our model to deal with this situation by assigning probabilities to sets of transitions, rather than just individual transitions. We have chosen to consider only discrete probability distributions here for ease of exposition.

at all meaningful. Nevertheless, the distributions are different; depending on the choice we make, we can be led to quite different analyses of a protocol.

To understand this distinction, consider the Coordinated Attack problem [Gra78]. Two generals A and B must decide whether to attack a common enemy, but we require that any attack be a coordinated attack; that is, A attacks iff B attacks. Unfortunately, they can communicate only by messengers who may be captured by the enemy. It is known that it is impossible for the generals to coordinate an attack under such conditions [Gra78, HM84]. Suppose, however, we relax this condition and require only that the generals coordinate their attack with high probability [FH88, FZ88]. To eliminate all nondeterminism, let us assume general A flips a fair coin to determine whether to attack, and let us assume the probability a messenger is lost to the enemy is 1/2. Our new correctness condition is that the condition "A attacks iff B attacks" holds with probability .99.

Consider the following two-step solution CA_1 to the problem. At round 0, A flips a coin and sends 10 messengers to B iff the coin landed heads. At round 1, B sends a messenger to tell A whether it has learned the outcome of the coin toss. At round 2, A attacks iff the coin landed heads (regardless of what it hears from B) and B attacks iff at round 1 it learned that the coin landed heads. It is not hard to see that if we put the natural probability space on the set of runs, then with probability at least .99 (taken over the runs) A attacks iff B attacks: if the coin lands tails then neither attacks, and if the coin lands heads then with probability at least .99 at least one of the ten messengers sent from A to B at round 0 avoids capture and both generals attack.

This is very different, however, from saying that at all times both generals know that with probability at least .99 the attack will be coordinated. To see this, consider the state just before attacking in which A has decided to attack but has received a message from B saying that B has not learned the outcome of the coin toss. At this point, A is certain the attack will not be coordinated. Although we have not yet given a formal definition of how to compute an agent's probability at a given point, it seems unreasonable for an agent to believe with high probability that an event will occur when information available to the agent guarantees it will not occur.

On the other hand, consider the solution CA_2 differing from the preceding one only in that B does not try to send a messenger to A at round 1 informing A about whether B has learned the outcome of the coin toss. An easy argument shows that in this protocol, at all times both generals have confidence (in some sense of the word) at least .99 that the attack will be coordinated. Consider B, for example, after having failed to receive a message from A. B reasons that either A's coin landed tails and neither general will attack, which would happen with probability 1/2, or A's coin landed heads and all messengers were lost, which would happen with probability $1/2^{11}$; and hence the conditional probability that the attack will be coordinated given that B received no messengers from A is at least .99.

As the preceding discussion shows, in a protocol which has a certain property P with high probability taken over the runs, an agent may still find itself in a state where it knows perfectly well that P does not (and will not) hold. While correctness conditions P for problems arising in computer science have typically been stated in terms of a probability distribution on the runs, it might be of interest to consider protocols where an agent knows P with high probability at all points. As we shall show, the probability distribution on the runs typically corresponds to each agent's probability distribution at time 0. Thus, we can view the probability on the runs as an *a priori* probability distribution. To require a fact to hold with high probability from each agent's point of view at all times is typically a much stronger requirement than that of requiring it to hold with high probability over the set of runs. Arguably, in many cases, it is also a more natural requirement. We return to this point later in the paper.

5 Definitions of probabilistic knowledge

We want to make sense of statements such as "at the point c, agent p_i knows φ holds with probability α." The problem is that, although we typically have a well-defined probability distribution on the set of runs in each computation tree, in order to make sense of such statements we need a probability distribution on the points p_i considers possible at c. The reason we need a distribution on points and not just on runs is that many interesting facts are facts about points and not about runs. Consider, for example, the fact "the coin landed heads." If the coin is flipped many times in a run, this fact may be true at some points and not others. If we were willing to restrict our attention to facts about the run, then we could make do simply with a distribution on runs, but this would preclude the discussion of many interesting events in a system.

We begin by reviewing the general framework of [FH88] in which, given a particular assignment of probability spaces to points and agents, we can make

sense of such statements about an agent's probabilistic knowledge. The remainder of the paper will focus on the construction of appropriate probability assignments.

Define a *probability assignment* $\mathcal{P}$ to be a mapping from an agent p_i and point c to a probability space $\mathcal{P}_{i,c} = (S_{i,c}, \mathcal{X}_{i,c}, \mu_{i,c})$. Here $S_{i,c}$ is a set of points, $\mathcal{X}_{i,c}$ is the set of measurable subsets of $S_{i,c}$, and $\mu_{i,c}$ is a probability function assigning a probability to the sets in $\mathcal{X}_{i,c}$.[3] In most cases of interest, one can think of $S_{i,c}$ as a subset of the points agent p_i considers possible at c, and of $\mu_{i,c}$ as indicating the relative likelihood according to p_i that a particular point in $S_{i,c}$ is actually the current point c.

Given such an assignment, let $S_{i,c}(\varphi)$ be the set of the points in $S_{i,c}$ satisfying φ; that is, $S_{i,c}(\varphi) = \{d \in S_{i,c} : d \models \varphi\}$. It is natural to interpret $\mu_{i,c}(S_{i,c}(\varphi))$ as the probability φ is true, according to agent p_i at the point c. One problem with this interpretation, of course, is that the set $S_{i,c}(\varphi)$ is not guaranteed to be measurable, and hence $\mu_{i,c}(S_{i,c}(\varphi))$ is not guaranteed to be well-defined. In order to deal with this problem, we follow the approach of [FH88], and make use of inner and outer measures. Given a probability space $(S, \mathcal{X}, \mu)$, the *inner measure* μ_* and *outer measure* μ^* is defined by

$$\mu_*(S') = \sup\{\mu(T) : T \subseteq S' \text{ and } T \in \mathcal{X}\}$$
$$\mu^*(S') = \inf\{\mu(T) : T \supseteq S' \text{ and } T \in \mathcal{X}\}$$

for all subsets S' of S. Roughly speaking, the inner (resp. outer) measure of $S_{i,c}(\varphi)$ is the best lower (resp. upper) bound on the probability φ is true, according to p_i at c. It is easy to see that $\mu^*(T) = 1 - \mu_*(T^c)$ for any set T, where T^c is the complement of T. Given a probability assignment $\mathcal{P}$, we write $\mathcal{P}, c \models Pr_i(\varphi) \geq \alpha$ to mean $\mu_{i,c_*}(S_{i,c}(\varphi)) \geq \alpha$.[4] Note that we need the probability assignment $\mathcal{P}$ to make sense of Pr_i. We take $K_i^\alpha\varphi$ to be an abbreviation for $K_i(Pr_i(\varphi) \geq \alpha)$; thus $K_i^\alpha\varphi$ means that agent p_i knows that the probability of φ is at least α since $Pr_i(\varphi) \geq \alpha$ holds at all points p_i considers possible.

We now have all the definitions needed to give semantics to a logical language of knowledge and probability. In particular, the language of most interest to us in the remainder of this paper is the language $\mathcal{L}(\Phi)$ obtained by fixing a set Φ of primitive propositions and closing under the standard boolean connectives (conjunction and negation), the knowledge operators K_i, probability formulas of the form $Pr_i(\varphi) \geq \alpha$, and the standard (linear time) temporal logic operators next $\bigcirc$ and until $\mathcal{U}$. Note that $\mathcal{L}(\Phi)$ is sufficiently powerful to express the operators K_i^α and the temporal operators henceforth $\Box$ and eventually $\Diamond$. In the context of a given system, we say that $\mathcal{L}(\Phi)$ is *state-generated* if each of the primitive propositions in Φ is a fact about the global state; and we say that $\mathcal{L}(\Phi)$ is *sufficiently rich* if for every countable set G of global states contained in a single computation tree there is a primitive proposition in Φ true at precisely those points with global states in G. This condition ensures, for example, that the language $\mathcal{L}(\Phi)$ is rich enough to allow us to talk about individual global states. The assumption that $\mathcal{L}(\Phi)$ is state-generated is quite reasonable in practice: we typically take the primitive propositions to represent facts about the global state, like "the coin landed heads," "the message was received," or "the value of variable x is 0." Sufficient richness is a technical condition required for a few of our results. We can always make a language sufficiently rich by adding primitive propositions.

We now have a natural way of making sense of knowledge and probability, given a probability assignment $\mathcal{P}$. Unfortunately, we still do not know how to choose $\mathcal{P}$, but our choices are somewhat more constrained than they may at first appear. We are given the computation trees and the associated distributions on runs, and we clearly want the distribution on the sample space $S_{i,c}$ of points we associate with agent p_i at point c to be related somehow to these distributions on runs. We next show that once we choose the sample spaces $S_{i,c}$, there is a straightforward way to use the distribution on runs to induce a distribution on $S_{i,c}$. Thus, once we are given an appropriate choice of sample spaces and the distributions on runs of the computation trees, we can construct the probability assignment. The problem of choosing a probability assignment, therefore, essentially reduces to choosing the sample spaces. This reduction will clarify important issues in determining the appropriate choice of probability assignments.

The idea of our construction is quite straightforward: given a sample space $S_{i,c}$ and a subset $S \subseteq S_{i,c}$, the probability of S (relative to $S_{i,c}$) is just the probability of the runs going through S normalized by the probability of the set of runs going through $S_{i,c}$. In other words, the probability of S is the conditional probability a run passes through S, given that the run passes through $S_{i,c}$.

In order for this simple idea to work, however, the set $S_{i,c}$ must satisfy a few requirements. One natural choice for $S_{i,c}$ is the set $\mathcal{K}_i(c)$ of all points agent p_i considers possible at c. In general, however, this set contains points from many different computation

[3] We often identify the probability space $\mathcal{P}_{i,c}$ with the sample space $S_{i,c}$; the intention should be clear from context.

[4] We remark that we can easily extend these definitions to more complicated formulas such as $Pr_i(\varphi) \geq 2Pr_i(\psi)$; see [FH88].

trees, and attempting to impose a distribution on this set of points leads to the same difficulties that led us to factor out nondeterminism and view a system as a collection of computation trees in the first place. Recall the example from Section 3 in which p_1 tosses a fair or biased coin, depending on whether his input is 0 or 1. Before (and after) the coin is tossed, p_2 considers four worlds possible, one from each possible run. We can no more place a probability on these points than we could place a probability on the four runs. On the other hand, given a point c from a run with input bit 1 (corresponding to the biased coin), if we restrict $S_{2,c}$ to consist of the two points in the computation tree with input 1, then we can put a probability on the two points in the obvious way and compute the probability of heads as 2/3. This intuition leads us to require that each set $S_{i,c}$ be contained entirely within a single computation tree:

REQ_1. *All points of* $S_{i,c}$ *are in* $\mathcal{T}(c)$.

We remark that, while REQ_1 does not allow us to take $S_{i,c}$ to be all of $\mathcal{K}_i(c)$, it still seems natural to choose $S_{i,c} \subseteq \mathcal{K}_i(c)$. We say that a probability assignment is *consistent* if it satisfies this condition. As pointed out in [FH88], a consequence of this is that if p_i knows φ, then φ holds with probability 1; that is, $K_i(\varphi) \Rightarrow (Pr_i(\varphi) = 1)$.[5] With a consistent assignment, it cannot be the case that agent p_i both knows φ and at the same time assigns $\neg\varphi$ positive probability.

The single condition REQ_1, however, is not enough for our idea for imposing a distribution on the set $S_{i,c}$ of points to work. Because this idea involves conditioning on the set of runs passing through $S_{i,c}$, the definition of conditional probability forces us to require that that this set of runs is a measurable set with positive measure. Suppose $\mathcal{T}(c) = (\mathcal{R}_A, \mathcal{X}_A, \mu_A)$, for some adversary A. Given a set S of points contained in $\mathcal{T}(c)$, denote by $\mathcal{R}(S)$ the set of runs passing through S; that is, $\mathcal{R}(S) = \{r \in \mathcal{R}_A : (r,k) \in S \text{ for some } k\}$. We require that

REQ_2: $\mathcal{R}(S_{i,c}) \in \mathcal{X}_A$ *and* $\mu_A(\mathcal{R}(S_{i,c})) > 0$.

REQ_2 is a relatively weak requirement. The following lemma shows that, in practice, REQ_2 is typically satisfied. A set S of points is said to be *state-generated* if $(r,k) \in S$ and $r(k) = r'(k')$ imply $(r',k') \in S$; in other words, S contains all points with the same global state as (r,k).

Proposition 1: If $S_{i,c}$ is state-generated, then $S_{i,c}$ satisfies REQ_2.

[5] In fact, as pointed out in [FH88], this axiom characterizes the property that the probability space used by p_i is a subset of the points that p_i considers possible.

Given a set of points $S_{i,c}$ satisfying REQ_1 and REQ_2, we now make precise our idea for imposing a distribution on $S_{i,c}$. Intuitively, to construct the collection $\mathcal{X}_{i,c}$ of measurable subsets of $S_{i,c}$, we project the measurable subsets of the runs of $\mathcal{T}(c)$ onto $S_{i,c}$. Formally, given a set $\mathcal{R}'$ of runs and a set S of points, we define $Proj(\mathcal{R}', S) = \{(r,k) \in S : r \in \mathcal{R}'\}$. We define

$$\mathcal{X}_{i,c} = \{Proj(\mathcal{R}', S_{i,c}) : \mathcal{R}' \in \mathcal{X}_A\}.$$

Finally, we define the probability function $\mu_{i,c}$ on the measurable subsets of $S_{i,c}$ via conditional probability:

$$\mu_{i,c}(S) = \mu_A(\mathcal{R}(S) \mid \mathcal{R}(S_{i,c})) = \frac{\mu_A(\mathcal{R}(S))}{\mu_A(\mathcal{R}(S_{i,c}))}$$

for all $S \in \mathcal{X}_{i,c}$. Let $P_{i,c} = (S_{i,c}, \mathcal{X}_{i,c}, \mu_{i,c})$.

Proposition 2: If $S_{i,c}$ satisfies REQ_1 and REQ_2, then $P_{i,c}$ is a probability space.

We can now formalize our intuition that the construction of probability assignments reduces to the choosing of sample spaces. Define a *sample space assignment* to be a function $\mathcal{S}$ that assigns to each agent p_i and point c a sample space $\mathcal{S}(i,c) = S_{i,c}$ satisfying REQ_1 and REQ_2. Define a *transition probability assignment* to be a function τ mapping an adversary A to the distribution μ_A on the runs of the computation tree $\mathcal{T}_A$ (corresponding to adversary A) induced by the transition probabilities labeling the edges of $\mathcal{T}_A$. Given assignments $\mathcal{S}$ and τ, our construction shows how to obtain a probability space $\mathcal{P}_{i,c}$ for all agents p_i and all points c. This naturally determines a probability assignment $\mathcal{P}$, which we call the the *probability assignment induced by* $\mathcal{S}$ *and* τ. We omit reference to τ when it is clear from context. For future reference, we define a fact φ to be *measurable with respect to* $\mathcal{S}$ if $S_{i,c}(\varphi) \in \mathcal{X}_{i,c}$ for all agents i and points c.

The preceding discussion makes precise the idea that choosing a probability assignment reduces to choosing a sample space assignment, but still does not help us choose the sample space assignment. Different choices result in probability assignments with quite different properties. Let us return to the example in the introduction, where p_1 tosses a fair coin, and neither p_2 nor p_3 observe the outcome. Clearly, at time 2 (after the coin has been tossed), p_2 considers two points possible: say h (the coin landed heads) and t (the coin landed tails). Consider the sample space assignment $\mathcal{S}^1$ such that $\mathcal{S}^1(2,h) = \mathcal{S}^1(2,t) = \{h,t\}$. Thus, at both of the points h and t, the same sample space is being used. In this case, at both points, the probability of heads is 1/2. Thus, with respect

to the induced probability assignment, p_2 knows that the probability of heads is 1/2. On the other hand, consider assignment $\mathcal{S}^2$ such that $\mathcal{S}^2(2,h) = \{h\}$ and $\mathcal{S}^2(2,t) = \{t\}$. With respect to the induced probability assignment, the probability of heads at h according to p_2 is 1, while the probability of heads at t is 0. In this case, all that p_2 can say is that it knows that the probability of heads is either 1 or 0, but it doesn't know which. Which is the right probability assignment? As we hinted in the introduction, the answer depends on another type of adversary, the one that p_2 views itself as playing against. This is the focal point of the next section.

We conclude this section with one further example. Consider a system where a fair die is tossed by p_1 and p_2 does not know the outcome. Suppose that at time 2 the die has already been tossed. Let $c_1, \ldots c_6$ be the six points corresponding to the possible outcomes of the die. What sample space assignment should we use for p_2. One obvious choice is to take the assignment $\mathcal{S}^1$ which assigns the same sample space at all six points, the space consisting of all the points. With respect to this sample space, each point will have probability 1/6. Let φ be the statement "the die landed on an even number." Clearly, in the probability space induced by this sample space, φ holds with probability 1/2. Since p_2 uses the same sample space at all six points, agent p_2 knows that the probability of φ is 1/2. A second possibility is to consider two sample spaces $S_1 = \{c_1, c_2, c_3\}$ and $S_2 = \{c_4, c_5, c_6\}$; let the assignment $\mathcal{S}^2$ assign the sample space S_1 to agent p_2 at all the points in S_1, and the sample space S_2 at all the points in S_2. Thus, at all the points in S_1, the probability of φ is 1/3, while at all the points in S_2, the probability of φ is 2/3. All p_2 can say is that it knows that the probability of φ is either 1/3 or 2/3, but it does not know which.

Clearly we can subdivide the six points into even smaller subspaces. It is not too hard to show that the more we subdivide, the less precise is p_2's knowledge of the probability. (We prove a formal version of this statement in the next section.) But why bother subdividing? Why not stick to the first sample space assignment, which gives the most precise (and seemingly natural) answer? Our reply is that, again, this may not be the appropriate answer when playing against certain adversaries.

6 Probability assignments in synchronous systems

We first consider the problem of selecting appropriate probability assignments in completely synchronous systems. Intuitively, a system is synchronous if all agents effectively have access to a global clock. Formally, a system is *synchronous* [HV89] if for all points (r,k) and (r',k') and all agents p_i, if $r_i(k) = r'_i(k')$ then $k = k'$. This means, for example, that no two points an agent p_i considers indistinguishable can lie on the same run.

When considering probability, it turns out that many things become much easier in the context of synchronous systems. For example, it turns out that, in practice, sample space assignments satisfy three natural properties: (a) they are state-generated; (b) they satisfy the property that $c \in S_{i,c}$ for all agents p_i and points c; and (c) they are *uniform*, which means that $d \in S_{i,c}$ implies $S_{i,d} = S_{i,c}$ for all agents p_i and points c and d.[6] We say that $\mathcal{S}$ (and its induced probability assignment) is *standard* if it satisfies these three properties. For the remainder of this section we restrict attention to standard assignments.

One convenient feature of synchronous systems is that all facts of interest are measurable. Recall that a $\mathcal{L}(\Phi)$ is state-generated with respect to a system $\mathcal{R}$ if all the primitive propositions in Φ are facts about the global state.

Proposition 3: In a synchronous system, if $\mathcal{S}$ is a consistent sample space assignment and $\mathcal{L}(\Phi)$ is state-generated, then φ is measurable with respect to $\mathcal{S}$ for all facts $\varphi \in \mathcal{L}(\Phi)$.

This result says that for all practical purposes we do not have to concern ourselves with nonmeasurable sets and inner measures in synchronous systems.

We begin by defining four probability assignments. Each of these assignment can be understood in terms of a betting game against an appropriate adversary. (This is the second type of adversary mentioned in the introduction.) We make this intuition precise after we have defined the probability assignments.

The first of these assignments corresponds to what decision theorists would call an agent's *posterior* probability. This is essentially the probability an agent would assign to an event given everything the agent knows. This intuitively corresponds to the bet an agent would be willing to accept from a copy of itself, someone with precisely the same knowledge that it has. We make this relationship between probability and betting precise shortly.

What probability space corresponds to an agent's conditioning on its knowledge in this way? Since we

[6] This condition is essentially the definition of a uniform probability assignment from [FH88]. A probability assignment induced by a uniform sample space assignment as we have defined it here is a uniform probability assignment in the sense of [FH88].

have identified an agent p_i's knowledge with the set of points p_i considers possible at c, this set of points seems the most natural choice for the space. As we have seen, however, this set of points is not in general contained in one computation tree. Thus, we consider instead the set of points in c's computation tree $\mathcal{T}(c)$ that p_i considers possible at c. This is just the set $Tree_{i,c} = \{d \in \mathcal{T}(c) : c \sim_i d\}$. It is clear that $Tree_{i,c}$ satisfies REQ_1; that it satisfies REQ_2 follows by Proposition 1 since it is state-generated. By Proposition 2, therefore, the induced probability space $(Tree_{i,c}, \mathcal{X}_{i,c}, \mu_{i,c})$ is indeed a probability space. Let $\mathcal{S}^{post}$ be the sample space assignment that assigns the space $Tree_{i,c}$ to agent p_i at the point c, and let $\mathcal{P}^{post}$ be the probability assignment induced by $\mathcal{S}^{post}$.

The probability space $\mathcal{P}^{post}_{i,c}$ has a natural interpretation. It is generated by conditioning on everything p_i knows at the point c and the fact that it is playing against the adversary A that generated the tree $\mathcal{T}_A$ in which c lies. Of course, the agent considers many adversaries possible. Thus, the statement $\mathcal{P}^{post}, c \models K_i^{\alpha}\varphi$ means that for all adversaries p_i considers possible at c (given its information at c), the probability of φ given all p_i knows is at least α. $\mathcal{P}^{post}$ is precisely the assignment advocated in [FZ88] in the synchronous case.

Suppose now that p_i were considering accepting a bet from someone (not necessarily an agent in the system) with complete knowledge of the past history of the system. In this case, we claim that the appropriate choice of probability space for p_i at the point $c = (r, k)$ is all the other points (r', k) that have the same prefix as (r, k) up to time k; in other words, all points with the global state $r(k)$. Call this set of points $Pref_{i,c}$. Note that $Pref_{i,c}$ is independent of p_i, and depends only on the point c. Moreover, $Pref_{i,c}$ is clearly state-generated (by $r(k)$ itself), so by Proposition 1 we can again use the construction of Proposition 2 to induce a natural probability distribution on this set of points. Let $\mathcal{S}^{fut}$ denote the sample space assignment that assigns $Pref_{i,c}$ to p_i at c, and let $\mathcal{P}^{fut}$ denote the probability assignment induced by $\mathcal{S}^{fut}$. We remark that this is the probability assignment used in [HMT88], as well as [LS82].

In the probability space $\mathcal{P}^{fut}_{i,c}$, any event that has already happened by the point c will have probability 1. Future events (that get decided further down the computation tree) still have nontrivial probabilities, which is why we have termed it a future probability assignment.

Let us reconsider yet again the coin tossing example from the introduction, where agent p_2 tosses a fair coin at time 1 but agents p_1 and p_3 do not learn the outcome. Since the coin has already landed at time 2, it is easy to check that we have $\mathcal{P}^{fut}, c \models K_1(Pr_1(heads) = 1 \vee Pr_1(heads) = 0)$. On the other hand, we have $\mathcal{P}^{post}, c \models K_1(Pr_1(heads) = 1/2)$. Thus, $\mathcal{P}^{post}$ and $\mathcal{P}^{fut}$ correspond to the two natural answers we considered for the probability of heads. They capture the intuition that the answer depends on the knowledge of the adversary p_1 is betting against: $\mathcal{P}^{fut}$ corresponds to betting against p_2, and $\mathcal{P}^{post}$ corresponds to betting against p_3.

Notice that in both the cases of $\mathcal{P}^{post}$ and $\mathcal{P}^{fut}$, the probability space associated with an agent at a point corresponds to the set of points the agent and the adversary *both* consider possible. Suppose, in general, that p_i is considering what an appropriate bet to accept from p_j would be. We claim that in this case the probability assignment should be generated by the *joint* knowledge of agents p_i and p_j, as represented by the intersection of the points they both consider possible; that is, by the set $Tree^j_{i,c} = Tree_{i,c} \cap Tree_{j,c}$. (Note that $Tree^i_{i,c} = Tree_{i,c}$, so that this construction can be viewed as a generalization of the previous one.) Again it is easy to see that $Tree^j_{i,c}$ is state-generated, so by Proposition 1 we can use Proposition 2 to induce the natural distribution on this set of points. Let $\mathcal{S}^j$ be the sample space assignment that assigns $Tree^j_{i,c}$ to p_i at c, and let $\mathcal{P}^j$ be the probability assignment induced by $\mathcal{S}^j$.

All the examples we have seen up to now have had the property that $S_{i,c} \subseteq \mathcal{K}_i(c)$, which means they are consistent. While consistency is a natural restriction on probability assignments, it is not a requirement of our framework. There may be be technical reasons for considering inconsistent assignments. For example, one obvious (although inconsistent) probability assignment associates with the point (r, k) the set of *all* time k points in its computation tree. Call this set $All_{i,c}$. ($All_{i,c}$ is in fact independent of p_i.) The probability space induced by the construction of Proposition 2 in this case simulates the probability on the runs. Let us denote the associated sample space and probability assignments by $\mathcal{S}^{prior}$ and $\mathcal{P}^{prior}$. Notice that if p_i uses the probability space $\mathcal{P}^{prior}_{i,c}$, it is essentially ignoring all that it has learned up to the point c, which is why we have termed it a prior probability.

All four of the sample space assignments we have constructed are standard assignments. It is not difficult to see, in fact, that any assignment constructed on the basis of some adversary's knowledge will be standard. This lends some justification to our restriction to standard assignments. We can view these four assignments as points in a lattice of all possible standard sample space assignments. We define an ordering $\leq$ on this lattice as follows: $\mathcal{S}' \leq \mathcal{S}$ iff every space $S_{i,c}$ of $\mathcal{S}$ can be partitioned into the spaces $S'_{i,d}$

of S' with $d \in S_{i,c}$. Consider $\mathcal{S}^{post}$ and $\mathcal{S}^{fut}$, for example. Every set $Tree_{i,c}$ of $\mathcal{S}^{post}$ can be partitioned into the sets $Tree^j_{i,d}$ of $\mathcal{S}^{fut}$ with $d \in Tree_{i,c}$. In fact, it is clear that

$$\mathcal{S}^{fut} \leq \mathcal{S}^j \leq \mathcal{S}^{post} \leq \mathcal{S}^{prior}.$$

Furthermore, notice that $\mathcal{S}^{post}$ is greatest (with respect to $\leq$) among all consistent sample space assignments.

Intuitively, $\mathcal{S}' \leq \mathcal{S}$ if the spaces of $\mathcal{S}'$ are a refinement of the spaces of $\mathcal{S}$. In the case of consistent assignments, if we interpret $S_{i,c}$ as the intersection of p_i's knowledge with the adversary's knowledge, $\mathcal{S}' \leq \mathcal{S}$ means the adversary corresponding to $\mathcal{S}'$ considers fewer points possible and hence knows more than the adversary corresponding to $\mathcal{S}$. This means, for example, that $\mathcal{S}^{post}$, as the maximal consistent assignment, corresponds to the least powerful adversary.

The ordering on sample spaces assignments induces an obvious ordering on probability assignments: if $\mathcal{S}' \leq \mathcal{S}$, then the induced assignments $\mathcal{P}'$ and $\mathcal{P}$ satisfy $\mathcal{P}' \leq \mathcal{P}$. An important point to note is that if $\mathcal{P}' \leq \mathcal{P}$, then $\mu'_{i,c}$ can be obtained from $\mu_{i,c}$ by conditioning with respect to $S'_{i,c}$:

Proposition 4: In a synchronous system, if $\mathcal{P}' \leq \mathcal{P}$, then for all agents p_i, all points c, and all measurable subsets $S' \in \mathcal{X}'_{i,c}$ of $S'_{i,c}$

(a) $S' \in \mathcal{X}_{i,c}$ (so that, in particular, $S'_{i,c}$ itself is a measurable subset of $S_{i,c}$),

(b) $\mu_{i,c}(S'_{i,c}) > 0$,

(c) $\mu'_{i,c}(S') = \mu_{i,c}(S'|S'_{i,c}) = \frac{\mu_{i,c}(S')}{\mu_{i,c}(S'_{i,c})}$.

It follows that any consistent probability assignment can be obtained from $\mathcal{P}^{post}$ by conditioning.

We are now able to make precise the sense in which $\mathcal{P}^{post}$, $\mathcal{P}^j$, and $\mathcal{P}^{fut}$ are the "right" probability assignments for an agent to use when playing against an opponent who knows exactly as much as it does, when playing against p_j, and when playing against an opponent who has complete information about the past. We focus on $\mathcal{P}^j$ here, but the arguments are the same in all cases.

Consider the following betting game between agents p_i and p_j at a point c. Agent p_j offers p_i a payoff of β for a bet on φ. Agent p_i either accepts or rejects the bet. If p_i accepts the bet, p_i pays one dollar to p_j, and p_j pays β dollars to p_i if φ is true at c. It follows that p_i's *profit* on this bet is either $\beta - 1$ or -1 depending on whether φ is true or false at c when p_i accepts the bet, and 0 when p_i rejects the bet.

Intuitively, assuming that p_i is risk neutral,[7] p_i can always be convinced to accept a bet on φ no matter how low the probability of φ is, as long as p_i believes there is some nontrivial chance φ is true and the payoff β is high enough. Our intuition says there must be some relationship between the probability α with which p_i knows φ and this acceptable payoff β that would induce p_i to accept a bet on φ. If α is close to 0 then p_i might require a high payoff to make the bet's risk acceptable, but if α is close to 1 then p_i might be willing to accept a much lower payoff since the chance of losing is so remote. Our claim that $\mathcal{P}^j$ is the right probability assignment is based on the fact that $\mathcal{P}^j$ determines for an agent p_i the lowest acceptable payoff for a bet with p_j on a fact φ. In other words, $\mathcal{P}^j$ determines precisely how an agent p_i should bet when betting against p_j. In fact, $\mathcal{P}^j$ is in a sense the unique such probability assignment. We now make this intuition precise.

What should p_i consider an acceptable payoff for a bet on φ, assuming p_i does not want to lose money on the bet? Since p_j is presumably following some strategy for offering bets to p_i, the acceptable payoff should take this strategy into account. Consider, for example, the system in which p_j secretly flips a fair coin at time 0, and offers at time 1 to bet p_i that the coin landed heads. If p_j is following the strategy of always offering a payoff of \$2, independent of the outcome of the coin toss, then p_i can always safely accept the bet since, on average, it will not lose any money (that is, p_i's expected profit is zero). If p_j offers a payoff of \$2 only when the coin lands tails, then p_i is certain to lose money. On the other hand, if p_j offers a payoff of \$2 only when the coin lands heads, then it is p_j who is certain to lose money. While we expect that p_j will not follow a strategy that will cause it to lose money, we assume only that p_j's strategy for offering bets depends only on its local state. In other words, given two points p_j is unable to distinguish, p_j must offer the same payoff for a bet on φ at both points. Formally, a *strategy* for p_j is a function from p_j's local state at a point c to the payoff p_j should offer p_i for a bet on φ at c. Similarly, we assume that p_i's strategy for accepting or rejecting bets (that is, for computing acceptable payoffs) is also a function of its local state.

Again, what should p_i consider an acceptable payoff for a bet on φ? Suppose p_i decides it will accept any bet on φ with a payoff of at least $1/\alpha$ when its

[7] Informally, an agent is said to be risk neutral if it is willing to accept all bets where its expectation of winning is nonnegative.

local state is s_i (remember that p_i's strategy for accepting bets must be a function of its local state). Denoting by $Bet(\varphi, \alpha)$ the rule "accept any bet on φ with a payoff of at least $1/\alpha$," how well does p_i do by following $Bet(\varphi, \alpha)$ when its local state is s_i? Clearly p_i will win some bets and lose others, so we are interested in computing p_i's expected profit. This in turn depends on p_j's strategy. This leads us to compute, for each of p_j's strategies f, agent p_i's expected profit when p_i follows $Bet(\varphi, \alpha)$ and p_j follows f. Intuitively, if, for each of p_j's strategies f, agent p_i's expected profit is nonnegative, then p_i does not lose money on average by following $Bet(\varphi, \alpha)$, regardless of p_j's strategy.

Before we can compute p_i's expected profit, however, there is an important question to answer: What probability space should we use to compute this expectation at a point c? One reasonable choice is to take $Tree_{i,c}$; this would correspond to computing this expectation with respect to everything p_i knows. Another reasonable choice would be to take $Tree^j_{i,c}$. The intuition would be that p_i wants to do well for every possible choice of what p_j could do to p_i. The sets $Tree^j_{i,c}$ correspond to the different things p_j could do, since p_j's strategy is a function of its local state. For definiteness, we take the expectation with respect to the probability space $Tree^j_{i,c}$ here, and then show that our results would not have been affected (at least in the synchronous setting) if we had chosen the space $Tree_{i,c}$ instead.

Let the value of the random variable $W_f = W_f(\varphi, \alpha)$ at a point d denote p_i's profit (or winnings) at d, assuming p_i is following $Bet(\varphi, \alpha)$ and p_j is following f. Assume that φ is measurable with respect to $\mathcal{S}^j$. Let $E_{i,c}[W_f] = E_{Tree^j_{i,c}}[W_f]$ denote the expected value of W_f with respect to the probability space $Tree^j_{i,c}$. We say p_i *breaks even* with $Bet(\varphi, \alpha)$ at c if $E_{i,c}[W_f] \geq 0$ for every strategy f for p_j. We say the rule $Bet(\varphi, \alpha)$ is *safe* for p_i at c if p_i breaks even with $Bet(\varphi, \alpha)$ at all points p_i considers possible at c.

To justify our definition of safe bets, we now prove that the definition remains unchanged if we take the expectation with respect to $Tree_{i,c}$ instead of $Tree^j_{i,c}$. We define $\text{Tree}^j_{i,c}$-*safe* to mean safe as defined above, and $\text{Tree}_{i,c}$-*safe* just as we defined safe, except that now we take the expectation with respect to $Tree_{i,c}$ instead of $Tree^j_{i,c}$.

Proposition 5: For all facts φ, all agents p_i, and all points c, the rule $Bet(\varphi, \alpha)$ is $Tree_{i,c}$-safe for p_i at c iff $Bet(\varphi, \alpha)$ is $Tree^j_{i,c}$-safe for p_i at c.

Our claim that $\mathcal{P}^j$ is the right probability assignment to use when playing against p_j is made concrete by the following result which states that $\mathcal{P}^j$ determines for every agent p_i precisely what bets are safe when betting against p_j.

Theorem 6: For all facts φ measurable with respect to $\mathcal{P}^j$, all agents p_i, and all points c, the rule $Bet(\varphi, \alpha)$ is safe for p_i at c iff $\mathcal{P}^j, c \models K_i^\alpha \varphi$.

This result of is the core of our paper. It says that that $\mathcal{P}^j$ determines precisely what bets are safe for p_i to accept. If, using the probability assignment $\mathcal{P}^j$, agent p_i knows the probability of φ is at least α, then p_i will at least break even betting on φ when the payoff is $1/\alpha$. On the other hand, if, using $\mathcal{P}^j$, agent p_i considers it possible that the probability of φ is less than α, then there is a strategy p_j can use that causes p_i to lose money betting on φ when the payoff is $1/\alpha$. In other words, $\mathcal{P}^j$ is the right probability assignment to use when betting against p_j.

While this theorem is stated only for *measurable* facts φ, remember that Proposition 3 assures us that facts of interest are typically measurable in synchronous systems. In fact, the same theorem holds even for nonmeasurable facts, once we define an appropriate notion of expectation for such facts; we defer those details to the full paper.

The proof of Theorem 6 depends only on the fact that $\mathcal{P}^j$ is induced by $\mathcal{S}^j$ and is actually independent of the particular transition probability assignment τ determining the distribution on runs. In this sense it is really $\mathcal{S}^j$ that is determining what bets are safe for p_i to accept. We can formalize this intuition that an arbitrary sample space assignment $\mathcal{S}$ determines safe bets as follows. We say $\mathcal{S}$ *determines safe bets* against p_j if the following condition holds:

$$\mathcal{P}, c \models K_i^\alpha \varphi \text{ implies } Bet(\varphi, \alpha) \text{ is safe for } p_i \text{ at } c$$

for all facts $\varphi \in \mathcal{L}(\Phi)$, all agents p_i, and all points c, and all probability assignments induced by $\mathcal{S}$ (and some transition probability assignment τ). The proof of Theorem 6 shows that $\mathcal{S}^j$ determines safe bets against p_j. It turns out that there are other assignments that determine safe bets against p_j, but if the language is sufficiently rich, so that there are a lot of possible events that can be bet on, then $\mathcal{S}^j$ enjoys the distinction of being the maximum such assignment.

Theorem 7: In a synchronous system,

(a) if $\mathcal{S} \leq \mathcal{S}^j$, then $\mathcal{S}$ determines safe bets safe bets against p_j.

(b) if $\mathcal{S}$ determines safe bets against p_j and $\mathcal{L}(\Phi)$ is sufficiently rich, then $\mathcal{S} \leq \mathcal{S}^j$.

We interpret Theorems 6 and 7 as providing strong evidence that $\mathcal{S}^j$ is the right sample space assignment, and hence that $\mathcal{P}^j$ is the right probability assignment, to use when playing against an adversary with p_j's knowledge. It says that the only way for p_i to be guaranteed it is using a safe betting strategy against p_j is by assuming the adversary is at least as powerful as p_j. Intuitively, the more powerful the adversary the less confident the agent is that it will be able to win a bet with this adversary, and the higher the payoff the agent will require before accepting a bet. Consequently, p_i is being unduly conservative if it takes a probability assignment that corresponds to an agent that is more powerful than p_j since it may pass up bets it should accept.

In the process of making this intuition precise, we can prove a theorem that gives us further insight into relationships between sample space assignments on the lattice. Recall that we have defined $K_i^\alpha\varphi$ to mean agent p_i knows α is a *lower* bound on the probability of φ. We can extend this definition to deal with intervals in a straightforward way. We would like to define $K_i^{[\alpha,\beta]}\varphi$ to mean $K_i(\alpha \leq Pr_i(\varphi) \leq \beta)$, which should mean agent p_i knows the probability of φ is somewhere between α and β. Since φ may not correspond to a measurable set, what we really mean is that the inner measure of φ is at least α and the outer measure is at most β. Since we interpret Pr_i as inner measure when φ does not correspond to a measurable set, and since $\mu^*(T) = 1 - \mu_*(T^c)$ for any set T, we can capture this intuition in terms of our language by interpreting $K_i^{[\alpha,\beta]}\varphi$ as an abbreviation for $K_i[(Pr_i(\varphi)) \geq \alpha) \wedge (Pr_i(\neg\varphi) \geq 1-\beta)]$. We can now prove the following.

Theorem 8: In a synchronous system, if $\mathcal{P}'$ and $\mathcal{P}$ are consistent probability assignments satisfying $\mathcal{P}' < \mathcal{P}$, then

(a) for every fact φ, every agent p_i, every point c, and all α, β with $0 \leq \alpha \leq \beta \leq 1$, we have

$$\mathcal{P}', c \models K_i^{[\alpha,\beta]}\varphi \text{ implies } \mathcal{P}, c \models K_i^{[\alpha,\beta]}\varphi,$$

(b) there exist a fact φ, an agent p_i, a point c, and α, β with $0 \leq \alpha \leq \beta \leq 1$ such that

$$\mathcal{P}', c \not\models K_i^{[\alpha,1]}\varphi \text{ and yet } \mathcal{P}, c \models K_i^{[\alpha,1]}\varphi$$

$$\mathcal{P}', d \not\models K_j^{[0,\beta]}\neg\varphi \text{ and yet } \mathcal{P}, d \models K_j^{[0,\beta]}\neg\varphi.$$

If $\mathcal{L}(\Phi)$ is sufficiently rich, then $\varphi \in \mathcal{L}(\Phi)$.

Part (a) shows that an agent's confidence interval does not increase in the presence of a more powerful adversary; part (b) shows that it might actually decrease. The formula φ from part (b) gives an example of a case that agent p_i might be unduly conservative by using an inappropriate probability assignment. Using $\mathcal{P}'$, agent p_i would reject bets on φ with payoff $1/\alpha$ even though it should be accepting all such bets.

Our results show that $\mathcal{P}^{post}$ has a special status among probability assignments. It is a maximum assignment among consistent assignments in the lattice with the $\leq$ ordering, and so, by Theorem 8, gives the sharpest bounds on the probability interval among all consistent probability assignments. In addition, any other consistent probability assignment can be obtained from $\mathcal{P}^{post}$ by a process of conditioning. Finally, $\mathcal{P}^{post}$ is the probability assignment that corresponds to what decision theorists seem to use when referring to an agent's subjective (or posterior) probability. However, as we have seen, $\mathcal{P}^{post}$ may not always be the "right" probability assignment to use. The right choice depends on the knowledge of the adversary offering us the bet in the system we wish to analyze. Although $\mathcal{P}^{post}$ may give a smaller interval than $\mathcal{P}^j$ (intuitively giving sharper bounds on an agent's belief a fact is true), if p_i uses the better lower bound from $\mathcal{P}^{post}$ as a guide to deciding what bet to accept from p_j, it may wind up losing money. In fact, it follows from Theorems 7 and 8 that $\mathcal{P}^j$ is the probability assignment that gives an agent the best interval and still guarantees a good betting strategy.

Even in cases where $\mathcal{P}^{post}$ is the "right" choice, it is not necessarily the probability we want to use in computations. It may not always be necessary to obtain the sharpest interval of confidence possible. A rough bound may be sufficient. Theorem 8 shows that proving a lower bound on an agent's confidence using a certain choice of probability space implies the same bound holds with any definition higher in the lattice. The advantage of using a probability assignment that lies lower in lattice is that, because the individual probability spaces are smaller, the computations may be simpler. When arguing about the level of confidence of an agent, it seems best to choose a definition as low in the lattice as possible to make the proof as simple as possible, but high enough to enable one to prove a sufficiently high level of confidence.

In this section we have seen how to understand and construct probability assignments for synchronous systems. In the full paper, we also consider asynchronous systems, where we find it necessary to consider a third type of adversary sketched in the introduction.

7 An application: coordinated attack

We now apply these different probability assignments to understanding probabilistic coordinated attack as defined in Section 4. In [HM84] it is shown that a state of knowledge called *common knowledge* is a necessary condition for coordinated attack. Intuitively, a formula φ is common knowledge if all agents know φ, all agents know all agents know φ, and so on *ad infinitum*. In the same paper it is shown that common knowledge of nontrivial facts cannot be attained in systems where there is no upper bound on message delivery time (and, in particular, in asynchronous systems), and hence that coordinated attack is not possible in such systems. We now examine the relationship between probabilistic common knowledge and probabilistic coordinated attack.

The definition of common knowledge [HM84] captures the intuition given above as follows. Given a set $G \subseteq \{p_1, \ldots, p_n\}$ of agents, we define *everyone in G knows* φ by $E_G \equiv \bigwedge_{p_i \in G} K_i\varphi$. Defining $E_G^k\varphi$ inductively by $E_G^0\varphi = \varphi$ and $E_G^k\varphi = E_G E_G^{k-1}\varphi$, we define φ *is common knowledge to G* by $C_G\varphi \equiv \bigwedge_{k>0} E_G^k\varphi$. It is not hard to prove that common knowledge satisfies the following statements [HM85]:

1. *the fixed point axiom*: $C_G\varphi \equiv E_G(\varphi \wedge C_G\varphi)$.

2. *the induction rule*: From $\psi \supset E_G(\psi \wedge \varphi)$ infer $\psi \supset C_G\varphi$.

The first statement says that $C_G\varphi$ is a fixed point of the equation $X \equiv E_G(\varphi \wedge X)$. In fact, it can be shown to follow from the induction rule that $C_G\varphi$ is the *greatest* fixed point, and thus is implied by all other fixed points of this equation [HM85].

By direct analogy, probabilistic common knowledge is defined in [FH88] as the greatest fixed point of the equation $X \equiv E_G^\alpha(\varphi \wedge X)$, where $E_G^\alpha \equiv \bigwedge_{p_i \in G} K_i^\alpha\varphi$.[8] It is easy to show that the definition of $C_G^\alpha\varphi$ satisfies the obvious analogues of the fixed point axiom and induction rule given above.

Now consider the probabilistic attack problem, and suppose φ is the fact "A attacks iff B attacks." In the original coordinated attack problem, since φ is true at all points, the induction rule implies $C_G\varphi$ holds at all points. Are there implementations of the probabilistic attack problem where $C_G^\alpha\varphi$ holds at all points? The answer depends on the choice of probability assignment. Stronger assignments yield stronger notions of probabilistic common knowledge which make stronger requirements of the implementation.

Consider the assignment $\mathcal{P}^{fut}$. Here the adversary offering an agent a bet knows the entire global state at every point. If there is any point where the attack is uncoordinated, then no run extending this point can satisfy φ. At this point φ holds with probability 0 (according to $\mathcal{P}^{fut}$), so it easily follows that $C_G^\alpha\varphi$ cannot hold. This says that an algorithm achieves probabilistic coordinated attack with respect to $\mathcal{P}^{fut}$ iff it achieves coordinated attack. Since coordinated attack is known to be unattainable in asynchronous systems, we cannot get probabilistic coordinated attack either with respect to such a strong adversary.

Next consider the assignment $\mathcal{P}^{post}$. Here the adversary offering the bet has precisely the same knowledge as the agent itself. Consequently, if it is possible to reach a point at which the agent can determine from its local state that no run extending the point can satisfy φ, the agent knows φ does not hold, and hence neither does $C_G^\alpha\varphi$. Consequently, our first implementation CA_1 of the probabilistic attack problem does not have the property that $C_G^\alpha\varphi$ holds at all points (with respect to $\mathcal{P}^{post}$), but our second implementation CA_2 does. This can be proved by first showing notice that $E_G^\alpha\varphi$ holds at all points (with respect to $\mathcal{P}^{post}$) and hence by the induction rule (taking the formula ψ in the rule to be *true*), so does $C_G^\alpha\varphi$.

Notice that with respect to any consistent probability assignment, if at some point an agent in G knows φ does not hold, then $C_G^\alpha\varphi$ cannot hold at this point (since $C_G^\alpha\varphi$ implies $E_G^\alpha\varphi$ by the fixed point axiom, while $K_i\neg\varphi$ implies $\neg E_G^\alpha\varphi$ for all $i \in G$). Consequently, it cannot be the case that $C_G^\alpha\varphi$ holds at all points of CA_1 with respect to any consistent assignment. Is it possible for $C_G^\alpha\varphi$ to hold at all points of CA_1 with respect to *any* probability assignment? Since this algorithm guarantees φ holds with probability α, taken over the runs, the obvious solution is to make the assignment mimic the probability distribution on the runs. In particular, consider $\mathcal{P}^{prior}$. It is easy to see that with this assignment, every agent knows φ with probability α at all points of the system. Since $E_G^\alpha\varphi$ holds at all points, it follows by the induction rule that $C_G^\alpha\varphi$ holds at all points as well.

We summarize our discussion in the following proposition.

Proposition 9:

1. CA_1 achieves probabilistic coordinated attack with respect to $\mathcal{P}^{prior}$ but not $\mathcal{P}^{post}$.

[8] As is shown in [FH88], this definition is not equivalent to the infinite conjunction of $(E_G^\alpha)^k\varphi$, $k > 0$; however it is equivalent to the infinite conjunction of $(F_G^\alpha)^k\varphi$, $k > 0$, where we define $(F_G^\alpha)^k\varphi$ inductively by unwinding the fixed point equation: $(F_G^\alpha)^0\varphi = \varphi$ and $(F_G^\alpha)^k\varphi = E_G^\alpha(\varphi \wedge (F_G^\alpha)^{k-1}\varphi)$.

2. CA_2 achieves probabilistic coordinated attack with respect to $\mathcal{P}^{post}$ but not $\mathcal{P}^{fut}$.

3. A protocol achieves probabilistic coordinated attack with respect to $\mathcal{P}^{fut}$ iff it achieves coordinated attack, and hence no such protocol exists.

This proposition shows how increasing the power of the adversary (moving down in the lattice) strengthens the kind of guarantees that can be made for probabilistic attack. Note that all of the probability assignments agree at time 0, and the probability they assign to a set of points is identical to the probability of the set of runs going through those points. However, at later times, it is only $\mathcal{P}^{prior}$ that agrees with the initial probability on runs. Thus, for the other probability assignments, saying that φ holds with probability greater than α at all points (r,k) in $\mathcal{T}_A$ according to p_i will generally be a stronger statement than saying it holds with probability α taken over the runs of $\mathcal{T}_A$.

Of course, it is perfectly conceivable we might want to consider probability assignments besides those that we have discussed above, which will make yet more guarantees. Considering such intermediate assignments might be particularly appropriate in protocols where security is a major consideration, such as cryptographic protocols. There it becomes quite important to consider the knowledge of the agent we are betting against.

We remark that a slightly different definition of probabilistic coordinated attack is considered in [FZ88]: it is required only that the conditional probability both parties attack together, given that one of the parties attacks, is at least α.[9] It is then shown in [FZ88] that this form of probabilistic coordinated attack corresponds to all the agents having *average belief* of α that the attack will be coordinated. We can reinterpret these results in our language as showing that this notion of coordinated attack is equivalent to probabilistic common knowledge with respect to another probability assignment, much in the spirit of $\mathcal{P}^{prior}$. In particular, the probability space used by [FZ88] for this analysis is not $\mathcal{P}^{post}$, but an inconsistent probability assignment. However, it should be noted that one can be led to counterintuitive results using an inconsistent probability assignment. Consider $\mathcal{P}^{prior}$ in the context of CA_1. Since there is a point at which the information in agent A's local state guarantees the attack will not be coordinated, according to $\mathcal{P}^{prior}$ both $K_A^{\alpha}\varphi$ and $K_A\neg\varphi$ hold at this point. In other words, the choice of $\mathcal{P}^{prior}$ has the effect of saying that at a point an agent can have high confidence in a fact it knows to be false.

The preceding discussion raises another interesting point. While it is typically the case that in distributed systems applications only probabilities on runs are considered (which corresponds to $\mathcal{P}^{prior}$), it is not clear that this is always appropriate. If an agent running a probabilistic coordinated attack algorithm that is guaranteed to work with high probability over the runs finds itself in a state where it knows that the attack will not be coordinated, then it seems clear that it should not proceed with the attack. It may be worth reconsidering a number of algorithms to see if they can be redesigned to give stronger guarantees. This may be particularly appropriate in the context of zero-knowledge protocols [GMR89], where the current definitions allow a prover to continue playing against a verifier even if the prover knows perfectly will that it leak information. Although it is extremely unlikely that the prover will find itself in this situation, it may be worth trying to redesign the protocol to deal with this possibility. While *adaptive protocols*, where processors modify their actions in light of what they have learned, are common in the control theory literature, the probabilistic algorithms that are used in distributed systems typically are not adaptive. It seems that a number of algorithms can be converted to adaptive algorithms with relatively little overhead. We hope to study this issue more carefully in a future work.

8 Conclusion

We have provided a framework for capturing knowledge and probability in distributed systems. Our framework makes it clear that in order for an agent to evaluate the probability of a formula φ at a given point, it needs to specify the adversary (or, more accurately, adversaries) which determines the probability space. This use of adversaries may help clear up a number of subtle issues in the study of probability, such as what the probability that a coin lands heads is after the coin has been tossed. In addition, our approach allows us to unify the different approaches to probability in distributed systems that have appeared in earlier works. Of course, what needs to be done now is to use these definitions to analyze probabilistic (especially cryptographic) protocols!

[9] Although it is not clear from the definition of probabilistic attack given in [FZ88] over what the probability is being taken, the results given clearly assume that the probability is being taken over the runs.

Acknowledgments

Thanks to Moshe Vardi, Hagit Attiya, and Adam Grove for stimulating discussions.

References

[BG54] D. Blackwell and M. A. Girshick, *Theory of Games and Statistical Decisions*, John Wiley & Sons, Inc., New York, 1954.

[FH88] R. Fagin and J. Halpern, Reasoning about knowledge and probability: preliminary report, *Proceedings of the Second Conference on Theoretical Aspects of Reasoning about Knowledge* (M. Vardi, ed.), Morgan Kaufmann, 1988, pp. 277–293.

[FZ88] M. J. Fischer and L. D. Zuck, *Reasoning about Uncertainty in Fault-Tolerant Distributed Systems*, Technical Report YALEU/DCS/TR-643, Yale University, 1988.

[GMR89] S. Goldwasser, S. Micali, and C. Rackoff, The knowledge complexity of interactive proof systems, *SIAM Journal on Computing* **18**:1, 1989, pp. 186–208.

[Gra78] J. Gray, Notes on database operating systems, *Operating Systems: An Advanced Course* (R. Bayer, R. Graham, and G. Seegmuller, eds.), Lecture Notes in Computer Science, Vol. 66, Springer-Verlag, 1978. Also appears as *IBM Research Report RJ 2188*, 1987.

[HF88] J. Halpern and R. Fagin, Modelling knowledge and action in distributed systems, *Concurrency 88* (F. Vogt, ed.), 1988, pp. 18–32. A revised and expanded version appears as IBM Research Report RJ 6303, 1988.

[HM84] J. Halpern and Y. Moses, Knowledge and common knowledge in a distributed environment, *Proc. 3rd ACM Symp. on Principles of Distributed Computing*, 1984, pp. 50–61. A revised and expanded version appears as IBM Research Report RJ 4421, 1988.

[HM85] J. Halpern and Y. Moses, A guide to the modal logics of knowledge and belief, *Ninth International Joint Conference on Artificial Intelligence (IJCAI-85)*, 1985, pp. 480–490.

[HMT88] J. Halpern, Y. Moses, and M. Tuttle, A knowledge-based analysis of zero knowledge, *Proc. 20th ACM Symp. on Theory of Computing*, 1988, pp. 132–147.

[HV89] J. Halpern and M. Vardi, The complexity of reasoning about knowledge and time, I: Lower bounds, *Journal of Computer and System Sciences* **38**:1, 1989, pp. 195–237.

[Lew80] D. Lewis, A subjectivist's guide to objective chance, *Ifs* (W. L. Harper, R. Stalnaker, and G. Pearce, eds.), D. Reidel Publishing Company, 1980, pp. 267–297.

[LS82] D. Lehmann and S. Shelah, Reasoning about time and chance, *Information and Control* **53**, 1982, pp. 165–198.

[Mer85] N. D. Mermin, Is the moon there when nobody looks? Reality and the quantum theory, *Physics Today* **38**:4, 1985, pp. 38–47.

[Rab80] M. O. Rabin, Probabilistic algorithm for testing primality, *Journal of Number Theory* **12**, 1980, pp. 128–138.

[Rab82] M. O. Rabin, N-process mutual exclusion with bounded waiting by $4 \cdot \log n$-valued shared variable, *Journal of Computer and System Sciences* **25**:1, 1982, pp. 66–75.

[SS77] R. Solovay and V. Strassen, A fast Monte Carlo test for primality, *SIAM Journal on Computing* **6**:1, 1977, pp. 84–85.

[Var85] M. Y. Vardi, Automatic verification of probabilistic concurrent finite-state programs, *Proc. 26th IEEE Symp. on Foundations of Computer Science*, 1985, pp. 327–338.

[VF80] B. C. Van Fraassen, A temporal framework for conditionals and chance, *Ifs* (W. L. Harper, R. Stalnaker, and G. Pearce, eds.), D. Reidel Publishing Company, 1980, pp. 323–340.