Knowledge, Probability, and Adversaries

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

Mark R. Tuttle
Digital Equipment Corporation
Cambridge Research Lab
Cambridge, MA 02139
tuttle@crl.dec.com

December 21, 1991

Abstract: What should it mean for an agent to know or believe an assertion is true with probability .99? Different papers [FH94, FZ88a, HMT88] give different answers, choosing to use quite different probability spaces when computing the probability that an agent assigns to an event. We show that each choice can be understood in terms of a betting game. This betting game itself can be understood in terms of three types of adversaries influencing three different aspects of the game. The first selects the outcome of all nondeterministic choices in the system; the second represents the knowledge of the agent's opponent in the betting game (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 of this betting game. We conclude by showing how different assignments of probability spaces (corresponding to different opponents) yield different levels of guarantees in probabilistic coordinated attack.

1 Introduction

In nearly every field of research concerned with systems of interacting agents—be it distributed computing, artificial intelligence, or economics—people have found it useful to think about these systems in terms of knowledge. In game theory, for example, one player's strategy might change as it comes to know more about the other player's strategy. In all of these fields, an important subclass of interactions involve probability. For example, a player in a game might toss a coin in order to determine its next move. In such contexts, it is natural to find oneself reasoning—at least informally—about knowledge and probability and their interaction. This sort of reasoning is quite common in computer science, such as when reasoning about probabilistic primality-testing algorithms. Such an algorithm might guarantee that if the input n is a composite number, then with high probability the algorithm will find a "witness" that can be used to verify that n is composite. Loosely speaking, we reason, if an agent runs this algorithm on input n and the algorithm fails to find such a witness, then the agent knows that n is almost certainly prime, since the agent is guaranteed that the algorithm would almost certainly have found a witness had n been composite.

A number of recent papers have tried to formalize this sort of reasoning about knowledge and probability. Fagin and Halpern [FH94] present an abstract model for knowledge and probability in which they assign to each agent-state pair a probability space to be used when computing the probability, according to that agent at that state, that a formula φ is true.[1] In their framework, the problem of modeling knowledge and probability reduces to choosing this assignment of probability spaces. Although they show that more than one choice may be reasonable, they do not tell us how to make this choice. One particular (and quite natural) choice is made in [FZ88a] and some arguments are presented for its appropriateness;[2] another is made in [HMT88] and used to analyze interactive proof systems. It is not initially clear, however, which choice is most appropriate.

In this paper, we clarify the issues involved in choosing the right assignment of probability spaces. We argue that no single assignment is appropriate in all contexts: the right way to think about these assignments is in terms of strategies for a betting game, and different assignments can be viewed as most appropriate in the contexts of betting against different opponents in this game. Thinking in terms of probability, the truth of a statement such as "event E will occur with probability α" depends on the assignment of probability spaces. Thinking in terms of games, if the opponent can influence the occurrence of an event E in any way, then the truth of a statement such as "event E will occur with probability α" depends on the extent to which the opponent can influence E, and the success of any strategy depending on the occurrence of E depends on the power of the opponent. We establish a correspondence between assignments of probability spaces and powers of opponents, and hence establish a correspondence between assignments and winning strategies in this betting game against these opponents.

We find, however, there is more to the betting game than just the opponent you are betting against. Roughly speaking, the setting in which the game is played is also of great importance. We identify three aspects of the game and its environment that capture all that is relevant, and

[1] Related results about their model appear in [FH91, FHM90].

[2] In [FZ87, FZ88b], other definitions of knowledge involving probability are proposed, in order to analyze an interactive proof of quadratic residuosity, but these definitions are primarily concerned with accounting for the limited computational power of agents in the system.

model them in terms of three types of adversaries, each playing a fundamentally different role. We briefly describe these adversaries and their roles here, and explore them in greater depth in the rest of the paper.

When we analyze probabilistic protocols, we do so in terms of probability distributions on the *runs* or *executions* of the protocol. When we say that a protocol is correct with probability .99, we are trying to say that the protocol will do the right thing in .99 percent of the runs. In fact, a statement like ".99 percent of the runs" does not usually make sense, since we usually have probability distributions on subsets of the runs, but not on the entire set of runs. For example, consider any probabilistic primality-testing algorithm [Rab80, SS77]. For each fixed input (the number to be tested), the coins tossed during the algorithm induce a probability space on the set of runs of the algorithm with that input, but we do not have a distribution on the set of *all* runs because we are not willing to assume a distribution on the inputs. When we say that the algorithm works with probability .99, we really mean that for every choice of input the algorithm is correct in .99 of the runs with that input. The choice of input is a nonprobabilistic choice, and the coin tosses are probabilistic choices. The role of the first type of adversary in our framework is to distinguish between these two types of choices: this adversary factors out all nonprobabilistic choices in the system, so that for each adversary the remaining probabilistic choices induce a natural probability distribution on the set of runs with that adversary.

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_3 tosses a fair coin at time 0 and observes the outcome at time 1, but agents p_1 and p_2 never learn the outcome. What is the probability according to p_1 that the coin lands heads? Clearly at time 0, before the coin is tossed, it should be 1/2. What about at time 1? 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 of its having been tossed, 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, [van80, Lew81]). Interestingly, the same issue arises in quantum mechanics, in Schrödinger's famous cat-in-the-box thought experiment (see [Pag82] for a discussion).

We claim that these two choices of probability are best explained in terms of betting games (assuming honest players). At time 0, agent p_1 should certainly 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 (assuming p_1 is risk neutral[3]). Half the time the coin will land heads and p_1 will be \$1 ahead, and half the time the coin will land tails and p_1 will lose \$1, but on average p_1 will come out even. On the other hand, p_1 is clearly not willing to accept such an offer from p_3 at time 1 (since p_3 can tell at time 1 whether it is going to win the bet, and since p_3 is presumably willing to offer the bet only

[3]Informally, an agent is said to be risk neutral if it is willing to accept all bets where its expected winnings are nonnegative.

when it will win), although p_1 is still willing to accept this bet from p_2. The point here is that if you do not want to lose money 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 role played by the second type of 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. We sometimes identify the opponent with an adversary of this type. One obvious choice of opponent 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 [FZ88a]. 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 toss; 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 an agent does not even know exactly when the event to which it would like to assign a probability is being tested (for example, when a bet is being placed). Thus we need to consider a third type of adversary in asynchronous systems, whose role is to choose this time. To illustrate the need for this third type of adversary, 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 correspond to different adversaries choosing the times to perform the test in different ways. We remark that the case of asynchronous systems is also considered in [FZ88a]. We can understand the assignment of "confidence" made there as corresponding to playing against a certain class of adversaries of this third type.

As the preceding examples suggest, each of the earlier definitions of probabilistic knowledge that appear in the literature can be understood as the "best" definition for a particular choice of adversaries. On the other hand, we show that every choice of adversaries gives rise to a definition of probabilistic knowledge that is "best" for that choice. To make this precise, we formalize our intuition that the probability an agent assigns to an event is related to the payoff the agent is willing to accept in a betting game with an opponent. We define a particular betting game, and show that once we fix the choice of adversaries, there is a definition of probabilistic knowledge that is best in terms of doing as well as possible against an opponent whose knowledge is modeled by the second adversary. We show how this definition corresponds to a strategy that enables the agent to break even (at least) in the game, and how any other definition with this property corresponds to an overly conservative strategy that assumes the opponent is more powerful than it really is. 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 system of agents such as 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 opponent in the betting game. In Section 7, we consider asynchronous systems, where we also have to consider the third type of adversary. In Section 8 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. The paper ends with two appendices. In Appendix A we give the proofs of the results claimed in the paper, and in Appendix B we discuss some interesting secondary observations related to the rest of the paper.

2 Modeling systems

As the examples in the introduction show, the analysis of a probabilistic system typically depends on the choice of a probability distribution on the runs or executions of the system. In this section, we fix a model of computation that defines these runs, and in later sections we will vary the probability distributions associated with these runs. Our model is actually the model given in [HF89], which is itself a simplification of the model given in [HM90]. Both models are heavily influenced by models for distributed computation.

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. A global state is modeled as a tuple consisting of each process' local state, together with the state of the *environment* where, loosely speaking, the environment is intended to capture everything relevant to the state of the system that cannot be deduced from the agents' local states. Formally, a *global state* is an $(n+1)$-tuple $(s_e, s_1, \ldots, s_n)$ of local states, where s_i is the local state of agent p_i, and s_e is the state of the environment. A *run* of the system is mapping r from times to global states. We assume for sake of convenience that times are natural numbers. A *system* is a set $\mathcal{R}$ of runs, 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 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*. Later in the paper, we will assume that the entire history of a run r up to time k is encoded in the environment's state in $r(k)$. This allows us to think of the runs of a system in terms of a *computation tree*: nodes of the tree are global states (and correspond to a set of points), and the paths in the tree are the runs of the system. We say that a run r' *extends* a point (r,k) if r and r' pass through the same global states up to time k; that is, $r(k') = r'(k')$ for $0 \leq k' \leq k$.

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).[4] 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;

[4] 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 [HM90] for a complete formal treatment of the syntax and semantics of such a language.

that is, if $r_i(k) = r'_i(k')$. We use $\mathcal{K}_i(r,k)$ to denote $\{(r',k') \mid (r,k) \sim_i (r',k')\}$, the set of points agent p_i considers possible at (r,k). Following [HM90] (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 $(r',k') \in \mathcal{K}_i(r,k)$. 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 the first type of adversary sketched in the introduction.

Consider the simple system consisting of a single agent that tosses 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. The coin toss induces a very natural distribution on the two runs: each is assigned probability 1/2.

Now consider the system (suggested by Moshe Vardi; a variant appears in [FZ88a]) 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 tosses the fair coin once and halts. If the input bit is 1, p_1 tosses the biased coin and halts. This system consists of four runs of the form $\langle b, c\rangle$, where b is the value of the input bit and c is the outcome of the coin toss. What is the appropriate probability distribution on the runs of this system? For example, what is the probability of heads?

Clearly the *conditional* probability of heads given that the input bit is 0 should be 1/2, while the conditional probability of heads given the input bit is 1 should be 2/3. But what is the *unconditional* probability of heads? If we are given a distribution on the inputs, then it is easy to answer this question. If we assume, for example, that 0 and 1 are equally likely as input values, then we can compute that the probability of heads is $\frac{1}{2}\cdot\frac{1}{2} + \frac{1}{2}\cdot\frac{2}{3} = \frac{7}{12}$. If we are not given a distribution on the inputs, then the question has no obvious answer. It is tempting to assume, therefore, that such a distribution exists. Often, however, assuming a particular fixed distribution on inputs 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.

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. Moreover, assuming the existence of some unknown distribution on the inputs simply moves all problems arising from nondeterminism up one level. Although we have a distribution on the space of input values, we have no distribution on the space of probability 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 tossing example above, we would consider two probability spaces, one 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.[5]

We want to stress that although this example may seem artificial, analogous examples frequently arise in the literature. In a probabilistic primality-testing algorithm [Rab80, SS77], for example, we do not want to assume a probability distribution on the inputs. We want to know that for each choice of input, the algorithm gives the right answer with high probability. Rabin's primality-testing algorithm [Rab80] is based on the existence of a polynomial-time computable predicate $P_n(a)$ with the following properties: (1) if n is composite, then at least $3/4$ of the $a \in \{1, \dots, n-1\}$ cause $P_n(a)$ to be true, and (2) if n is prime, then no such a causes n to be true. Rabin's algorithm generates a polynomial number of a's at random. If $P_n(a)$ is true for any of the a's generated, then the algorithm outputs "composite"; otherwise it outputs "prime". Property (2) guarantees that if the algorithm outputs "composite", then n is definitely composite. If the algorithm outputs "prime", then there is a chance that n is not prime, but property (1) guarantees that this is very rarely the case: if n is indeed composite, then with high probability the algorithm outputs "composite". If the algorithm outputs "prime", therefore, it might seem natural to say that n is prime with high probability; but, of course, this is not

[5]Often, even in the presence of nondeterminism, we *can* impose a meaningful distribution on the runs of a system without factoring the system into subsystems, but the resulting distribution still may not capture all of our intuition. The problem in the preceding example is that probabilistic events (the coin toss) depend on nonprobabilistic events (the input bit). Suppose, however, the agent tosses a fair coin regardless of the input bit's value. Now it is natural to assign probability 1/2 to each of the events $\{(1,h),(0,h)\}$ and $\{(1,t),(0,t)\}$ that the coin lands head and tails, respectively. Consider, however, the situation (discussed in [FH94, HMT88]) where an agent performs a given action a iff the input bit is 1 and the coin landed heads, or the input bit is 0 and the coin landed tails. It is natural to argue that the probability the agent performs the action a is also 1/2: if the input bit is 1 then with probability 1/2 the coin will land heads and a will be performed; and if the input bit is 0 then with probability 1/2 the coin will land tails and a will be performed. Unfortunately, our "natural" distribution on the runs of the system does not support this line of reasoning, since this distribution does not assign a probability to the set $\{(1,h),(0,t)\}$ corresponding to the performance of a. In fact, if we could assign a probability to this set, then we would have to consider it a measurable set. Using this information, we could then prove that the sets $\{(1,h),(1,t)\}$ and $\{(0,h),(0,t)\}$ are also measurable sets. This means that we would have to assign a probability to having the input bit set to 0 or 1, but the setting of the input bit was assumed to be nondeterministic! Again, however, if we factor out this initial nondeterminism, we can view the system as two subsystems with obvious associated probability distributions, and within each subsystem the action a is performed with probability 1/2. This is precisely what the reasoning underlying our intuition is implicitly doing.

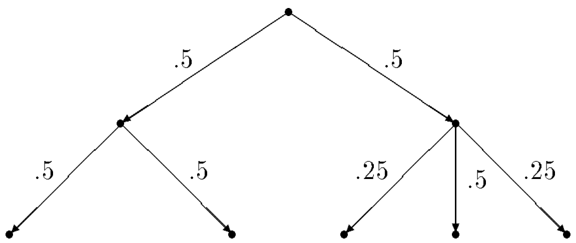

Figure 1: A (labeled) computation tree.

quite right. The input n is either prime or it is not; it does not make sense to say that it is prime with high probability. On the other hand, it does make sense to say that the algorithm gives the correct answer with high probability. The natural way to make this statement precise is to partition the runs of the algorithm into a collection of subsystems, one for each possible input, and prove that the algorithm gives the right answer with high probability in each of these subsystems, where the probability on the runs in each subsystem is generated by the random choices for a. While for a fixed composite input n there may be a few runs where the algorithm incorrectly outputs "prime", in almost all runs it will give the correct output.

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 tosses) 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 in this class the system satisfies some condition.

As we now show, if we view all nondeterministic choices as under the control of some adversary taken 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 [FZ88a].

Once we fix an adversary A, we can view the runs of the system with this adversary as a (labeled) *computation tree* $\mathcal{T}_A$ (see Figure 1). As in Section 2, nodes of the tree are global states and paths in the tree are runs. Now, however, 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. Intuitively, the value labeling an outgoing edge of node s represents the probability the system makes the corresponding transition from node s. 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.

It is natural to view this computation tree $\mathcal{T}_A$ 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 under countable union and complementation), and a probability function μ_A defined on sets in $\mathcal{X}_A$ so that the probability of the set of all 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 [FZ88a]), 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, in the case of finite runs, the probability of a run is just the product of the transition probabilities along the edges of the run.

It is occasionally useful to view this computation tree $\mathcal{T}_A$ as consisting of two components: the tree structure (that is, the unlabeled graph itself), and the assignment of transition probabilities to the edges of the tree. Given an unlabeled tree $\mathcal{T}_A$, we define a *transition probability assignment* for $\mathcal{T}_A$ to be a mapping τ assigning transition probabilities to the edges of $\mathcal{T}_A$. We will use the notation $\mathcal{T}_A$ at times to refer to the unlabeled tree, to the labeled tree, and to the induced probability space; which is meant should be clear from context.

We define a *probabilistic system* to consist of a collection of labeled computation trees (which we view as separate probability spaces), one for each adversary A in some set $\mathcal{A}$. 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 $\mathcal{A}$ of adversaries against which the system runs is typically made by the system designer when specifying correctness conditions for the system. 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 the order in which processors fail. One might also wish to restrict the computational power of the adversary to polynomial time. What is the most appropriate set of powers to assign to the adversary? It depends on the application.

4 Probability at a point

In the preceding section, we showed how to use the notion of an adversary to impose a meaningful probability distribution on the runs of a system. In order to define an agent's probabilistic knowledge at a given point, however, we seem to require a probability distribution on the points of the system, and not the runs. An agent's probability distribution on points must certainly be related to the distribution on runs if it is to be at all meaningful. Nevertheless, the two distributions may be quite different. For example, just as an agent's knowledge varies over time, we would expect the probability an agent assigns to an event to vary over time. In contrast, the distribution over runs can be viewed as a static distribution. Furthermore, depending on which of the two distributions we use, we can be led to quite different analyses of a protocol. The purpose of this section is to make clear the distinction between distributions over runs and distributions over points.

To begin, 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, HM90]. Suppose we relax this condition, however, and require only that the generals coordinate their attack with high probability [FH94, FZ88a]. To eliminate all nondeterminism, let us assume general A tosses a fair coin to determine whether to attack, and let us assume the probability a messenger is lost to the enemy is $1/2$. Let us assume further that the system is completely synchronous. 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 tosses 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 messages 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 (or a condition P) to hold with high probability from each agent's point of view at *all* times is typically a much stronger requirement than requiring it to hold with high probability over the set of runs. Arguably, in many cases, it is also a more natural requirement. It seems quite natural, for example, to require of a coordinated attack protocol that A have high confidence at all points that the attack will be coordinated, rather than allowing A to attack even when it is certain the attack will be uncoordinated.

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 most recent coin tossed landed heads". If a coin is tossed many times in a single run, say once every clock tick, this fact may be true at some points of the run and false at others, and hence is a fact about points and not about runs. When reasoning about probabilistic protocols, it seems quite natural to want to make formal statements of the form "agent p knows with probability $1/2$ that the most recent coin tossed by agent q landed heads"; this is not a fact about runs. If we restrict our attention to facts about runs, then we can make do simply with a distribution on runs, but this precludes (or at least complicates) the discussion of many interesting events in a system.

We begin by reviewing the general framework of [FH94] 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 $\mathcal{X}_{i,c}$.[6] 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 — that is, $S_{i,c} \subseteq \mathcal{K}_i(c)$ — and one can think 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.[7]

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

[6] We often follow the standard practice [Hal50, p. 73] of identifying the probability space $\mathcal{P}_{i,c}$ with the sample space $S_{i,c}$; the intention should be clear from context.

[7] Returning to the question of distributions on runs versus points, notice that as long as the set $S_{i,c}$ does not contain more than one point per run, there is a natural bijection from the probability on the points in $S_{i,c}$ to the probability on the runs going through $S_{i,c}$. In general, however, we allow more than one point on the same run to appear in $S_{i,c}$. As we shall see in the next section, this generality is useful when dealing with asynchronous systems.

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 [FH94], and make use of inner and outer measures. Given a probability space $(S, \mathcal{X}, \mu)$, the *inner measure* μ_* and *outer measure* μ^* are defined by

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

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 $\mu_{i,c_*}(S_{i,c}(\varphi)) \geq \alpha$.[8] 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 U. Note that $\mathcal{L}(\Phi)$ is sufficiently powerful to express the operators K_i^α and the temporal operators henceforth $\Box$ and eventually $\Diamond$.[9] 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 global state g there is a primitive proposition in Φ true at precisely those points with global state g. This condition ensures 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 such as "the coin landed heads", "the message was received", or "the value of variable x is 0". Each of these facts is a fact about the global state, since the history is recorded in the global state. 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,

[8] We remark that we can easily extend these definitions to more complicated formulas such as $Pr_i(\varphi) \geq 2Pr_i(\psi)$; see [FH94].

[9] We define $(r,k) \models \bigcirc\varphi$ iff $(r,k+1) \models \varphi$, so $\bigcirc\varphi$ is true at time k in a run iff φ is true at time $k+1$, after the next step. We define $(r,k) \models \varphi\, U\, \psi$ to mean there exists $\ell \geq k$ such that $(r,\ell) \models \psi$ and $(r,\ell') \models \varphi$ for all ℓ' with $k \leq \ell' < \ell$. Thus $\varphi\, U\, \psi$ is true at (r,k) if ψ is true at some point in the future, and φ is true until then. Recall that $\Diamond\varphi$, which says that φ is true at some point in the future, can be taken as an abbreviation of $\mathit{true}\, U\, \varphi$; and that $\Box\varphi$, which says that φ is true now and forever in the future, is an abbreviation for $\neg\Diamond\neg\varphi$.

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 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 its 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 [FH94], a consequence of this is that if p_i knows φ, then φ holds with probability 1; that is, $K_i(\varphi) \Rightarrow (Pr_i(\varphi) = 1)$.[10] 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.

In order to use the construction described above to impose a distribution on the set $S_{i,c}$, however, we must require more of $S_{i,c}$ than the single condition REQ_1. 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 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* $\mu_A(\mathcal{R}(S_{i,c})) > 0$.

Note that REQ_2 implies REQ_1. Nevertheless, REQ_2 is a relatively weak requirement. For example, the next result shows that REQ_2 is always satisfied in practice. A set S of points is

[10] In fact, as pointed out in [FH94], this axiom characterizes the property that the probability space used by p_i is a subset of the points that p_i considers possible.

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 and satisfies REQ_1, then $S_{i,c}$ satisfies REQ_2.

The proof of Proposition 1 (and all other technical results in this paper) can be found in Appendix A. We remark that this statement is actually independent of the transition probability assignment τ assigning probabilities to the edges of $\mathcal{T}_A$. While REQ_2 seems to depend on both $S_{i,c}$ and τ, Proposition 1 tells us we can choose $S_{i,c}$ without regard for τ and be confident REQ_2 will be satisfied for whatever τ we eventually choose, as long as $S_{i,c}$ is state generated.

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 choice of sample spaces. Given a system (i.e., a collection of labeled computation trees), 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. Given a sample space assignment $\mathcal{S}$, 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* $\mathcal{S}$. We note that the definition of $\mathcal{P}$ actually depends on *both* the sample space assignment $\mathcal{S}$ and the transition probability assignment τ (implicitly determined by the fact that we have labeled computation trees). There are times when it is convenient to start with an unlabeled computation tree, labeled by some transition probability assignment τ. In this case, we refer to $\mathcal{P}$ as the *probability assignment induced by* $\mathcal{S}$ *and* τ. 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 p_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_3 tosses a fair coin, and neither p_1 nor p_2 observe the outcome. Clearly, at time 1 (after the coin has been tossed), p_1 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 $\mathcal{S}^1(1,h) = \mathcal{S}^1(1,t) = \{h,t\}$. At both points h and t, the same sample space is being used; and at both points, with respect to the induced probability assignment, the probability of heads is 1/2. Thus, p_1 knows that the probability of heads is 1/2. According to this choice of sample spaces, p_1 has not learned anything about the outcome of the coin flip at time 1, and the probability of heads at time 1 is the same as at time 0. On the other hand, consider assignment $\mathcal{S}^2$ such that $\mathcal{S}^2(1,h) = \{h\}$ and $\mathcal{S}^2(1,t) = \{t\}$. With respect to the induced probability assignment, the probability of heads at h according to p_1 is 1, while the probability of heads at t is 0. According to this choice of sample spaces, the coin has either landed heads or landed tails, so all that p_1 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_1 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 $c_1, \dots c_6$ 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 are *inclusive*, which means $c \in S_{i,c}$ 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.[11] 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 consider only standard assignments.

One convenient feature of synchronous systems is that all facts of interest are measurable. Recall that $\mathcal{L}(\Phi)$ 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, standard 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 non-measurable sets and inner measures in synchronous systems. The proof is by induction on the structure of φ, and can be found in Appendix A.

We begin our examination of probability assignments in synchronous systems by defining four sample space assignments and their induced probability assignments. Each of these assignments can be understood in terms of a betting game against an appropriate opponent. (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 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 $\mathit{Tree}_{i,c} = \{d \in \mathcal{T}(c) : c \sim_i d\}$. It is clear that $\mathit{Tree}_{i,c}$ 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 $(\mathit{Tree}_{i,c}, \mathcal{X}_{i,c}, \mu_{i,c})$ is indeed a probability space. Let $\mathcal{S}^{post}$ be the sample space assignment that assigns the space $\mathit{Tree}_{i,c}$ 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 [FZ88a] 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,

[11] Condition (c) is essentially the definition of a uniform probability assignment from [FH94]. 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 [FH94].

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 Propositions 1 and 2 we can again induce a natural probability distribution on this set of points by conditioning on the runs passing through $Pref_{i,c}$. 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 whose outcome has already been determined before reaching the point c will have probability either 0 or 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 opponent 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 its opponent *both* consider possible. We can assume, without loss of generality, that this opponent is an agent in the system. Suppose, in general, that p_i is considering what an appropriate bet to accept from p_j would be. We claim (and show below) 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 $Tree^j_{i,c} = Tree_{i,c} \cap Tree_{j,c}$.[12] Again it is easy to see that $Tree^j_{i,c}$ is state generated, so by Propositions 1 and 2 we can induce the natural distribution on this set of points by conditioning on the runs passing through $Tree^j_{i,c}$. 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—$\mathcal{S}^{post}$, $\mathcal{S}^{fut}$, and $\mathcal{S}^j$—have had the property that $S_{i,c} \subseteq \mathcal{K}_i(c)$, which means they are consistent. As mentioned in Section 5, such assignments are characterized by the intuitively desirable condition $K_i(\varphi) \Rightarrow (Pr_i(\varphi) = 1)$; when we return to the coordinated attack problem in Section 8, we will see an example of an inconsistent assignment which causes an agent to know the attack will be coordinated with high probability, while

[12] Note that $Tree^i_{i,c} = Tree_{i,c}$. In this sense, this construction can be viewed as a generalization of the previous one, but the sample space assignment $\mathcal{S}^i$ being defined here is not the same as $\mathcal{S}^{post}$.

knowing that the attack will not be coordinated! While consistency seems a natural restriction on probability assignments, it is not a requirement of our framework. There may be be technical reasons for considering inconsistent assignments. One obvious (although inconsistent) probability assignment associates with the point $c = (r, k)$ 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—up to time k in r—which is why we have termed it a prior probability.

All four of the sample space assignments we have constructed are standard assignments. In fact, it is not difficult to see that any assignment constructed on the basis of some opponent'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 by $\mathcal{S}' \leq \mathcal{S}$ iff $S'_{i,c} \subseteq S_{i,c}$ for every agent p_i and point c. As usual, we write $\mathcal{S}' < \mathcal{S}$ when $\mathcal{S}' \leq \mathcal{S}$ for distinct $\mathcal{S}'$ and $\mathcal{S}$. An important property of this ordering is the following:

Proposition 4: If $\mathcal{S}$ and $\mathcal{S}'$ are standard assignments satisfying $\mathcal{S}' \leq \mathcal{S}$, then for every agent p_i and point c, the set $S_{i,c}$ can be partitioned into sets of the form $S'_{i,d}$ with $d \in S_{i,c}$.

Intuitively, this means that the sets $S'_{i,c}$ are refinements of the sets $S_{i,c}$, since the sets $S'_{i,c}$ are obtained by partitioning the sets $S_{i,c}$. 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.

In the case of consistent assignments, if we interpret $S_{i,c}$ as the intersection of p_i's knowledge with its opponent's knowledge, we can think of $\mathcal{S}' \leq \mathcal{S}$ as roughly meaning that the opponent corresponding to $\mathcal{S}'$ considers fewer points possible and hence knows more than the opponent corresponding to $\mathcal{S}$. This means, for example, that $\mathcal{S}^{post}$, as the maximal consistent assignment, corresponds to playing against the least powerful opponent.

The ordering on sample spaces assignments induces an obvious ordering on probability assignments: given two sample space assignments $\mathcal{S}'$ and $\mathcal{S}$ and their induced probability assignments $\mathcal{P}'$ and $\mathcal{P}$, respectively, we define $\mathcal{P}' \leq \mathcal{P}$ iff $\mathcal{S}' \leq \mathcal{S}$. Similarly, we define $\mathcal{P}' < \mathcal{P}$ iff $\mathcal{S}' < \mathcal{S}$. An important point to note is that if $\mathcal{P}'$ and $\mathcal{P}$ are consistent assignments satisfying $\mathcal{P}' \leq \mathcal{P}$, then $\mu'_{i,c}$ can be obtained from $\mu_{i,c}$ by conditioning with respect to $S'_{i,c}$:

Proposition 5: In a synchronous system, if $\mathcal{P}'$ and $\mathcal{P}$ are consistent, standard assignments satisfying $\mathcal{P}' \leq \mathcal{P}$, then for all agents p_i, all points c, and all measurable subsets $S' \in \mathcal{X}'_{i,c}$:

(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 in order to play the game, and p_j pays β dollars to p_i if φ is true at c. Thus, if p_i accepts this bet at the point c, then p_i's net gain is either $\beta - 1$ or -1 depending on whether φ is true or false at c; if p_i rejects the bet, we say its gain is 0.

One criterion of a good definition of probabilistic knowledge is that it should help p_i play this game. Intuitively, assuming that p_i is risk neutral and rational, there should be some relationship between the probability α with which p_i knows φ and the 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, while 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 tosses 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 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, 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. $Bet(\varphi,\alpha)$ is a safe bet for p_i against p_j.

To make this precise, fix some sample space assignment $\mathcal{S}$ and induced probability assignment $\mathcal{P}$, two agents p_i and p_j, and a fact φ that is measurable with respect to $\mathcal{S}$. 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. Let $E_{i,c}[W_f] = E_{\mathcal{S}_{i,c}}[W_f]$ denote the expected value of W_f with respect to the probability space $\mathcal{S}_{i,c}$. We say p_i *breaks even* with $Bet(\varphi,\alpha)$ with respect to $\mathcal{S}$ at c if $E_{i,c}[W_f] \geq 0$ for every strategy f for p_j. We say $Bet(\varphi,\alpha)$ is *$\mathcal{S}$-safe* for p_i at c if p_i *knows* that it breaks even with $Bet(\varphi,\alpha)$ with respect to $\mathcal{S}$ at c; that is, p_i breaks even with $Bet(\varphi,\alpha)$ at all points p_i considers possible at c.

There is still one important question left 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 $Tree^j_{i,c}$. This would correspond to the intuition that p_i wants to do well for every possible choice of what p_j could do to p_i: since p_j's strategy is a function of its local state, p_j's strategy could potentially have p_j offering different payoffs at points in the different sets $Tree^j_{i,c}$. It turns out that the choice doesn't make any difference (at least in the synchronous setting). The following result shows that the two definitions of safety are equivalent in synchronous systems.

Proposition 6: In a synchronous system, for all facts φ, all agents p_i and p_j, and all points c, the rule $Bet(\varphi,\alpha)$ is *Tree*-safe for p_i at c iff $Bet(\varphi,\alpha)$ is $Tree^j$-safe for p_i at c.

For the sake of being concrete, we choose the probability space $Tree^j_{i,c}$ since it is slightly easier to compute with. 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 7: 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 $\mathcal{P}^j$-safe for p_i at c iff $\mathcal{P}^j, c \models K_i^\alpha \varphi$.

We view this as the main result of our paper. It says 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 knows it 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 consider this notion in Appendix B.2

The proof of Theorem 7 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 as follows. We say that a standard sample space assignment $\mathcal{S}$ *determines safe bets* against p_j in a system consisting of unlabeled computation trees if, for all transition probability assignments τ assigning transition probabilities to edges of the computation trees, the following condition holds for the probability assignment $\mathcal{P}$ induced by $\mathcal{S}$ and τ:

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

for all facts $\varphi \in \mathcal{L}(\Phi)$, all agents p_i, and all points c. Notice that this definition quantifies over all transition probability assignments τ, requiring that the probability assignment induced by $\mathcal{S}$ determines safe bets regardless of the actual choice of τ. Our intuition says that the "right" way to go about constructing a probability assignment should not depend on the details of the transition probabilities. We would like some uniform way of choosing the probability space that does not change if there are small perturbations in the probability; Theorem 7 shows us that it is always possible to construct an assignment $\mathcal{P}^j$ in this way.

While the proof of Theorem 7 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. If the language $\mathcal{L}(\Phi)$ is sufficiently rich, however, 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 8: In a synchronous system, if $\mathcal{S}$ is a consistent, standard assignment, then

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

(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 7 and 8 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 opponent 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 opponent is at least as powerful as p_j. Intuitively, the more powerful the opponent the less confident the agent is that it will be able to win a bet with this opponent, 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.[13]

[13]Strictly speaking, we should justify the fact that p_i should use a rule of the form $Bet(\varphi, \alpha)$ in order to determine when to accept a bet. After all, why should such a simple threshold function be appropriate? It is conceivable that a better money-making strategy might tell p_i, say, to accept a bet on φ if the offered payoff is in the interval $[2, 5]$ or $[8, 10]$, and reject the bet otherwise. It is not hard to show, however, that because we make no assumption about the strategy being followed by p_j (other than requiring that it be a function of p_j's local state), this second strategy is safe for p_i at c iff it is safe for p_i at c to accept a bet on φ if the offered payoff is in the interval $[2, \infty)$, i.e. if $Bet(\varphi, 1/2)$ is safe for p_i at c. Consequently, an optimal strategy can always be taken to be a threshold function like $Bet(\varphi, \alpha)$.

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 $K_i[(Pr_i(\varphi)) \geq \alpha) \wedge (Pr_i(\neg\varphi) \geq 1 - \beta)]$. To relate this definition to our earlier definition of $K_i^\alpha\varphi$, notice that $K_i^\alpha\varphi$ is equivalent to $K_i^{[\alpha,1]}\varphi$. We can now prove the following.

Theorem 9: In a synchronous system, if $\mathcal{P}'$ and $\mathcal{P}$ are consistent, standard 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

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

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

If $\mathcal{L}(\Phi)$ is sufficiently rich, then $\varphi \in \mathcal{L}(\Phi)$.[14]

Part (a) shows that an agent's confidence interval does not increase in the presence of a more powerful opponent; part (b) shows that it might actually decrease. The fact 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.

In the light of these results, which probability assignment should we use in practice? 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 9, 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 opponent 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

[14] Recall from Section 2 that a fact is identified with a set of points, and that a formula is a sentence in a logic that denotes a set of points (that is, a formula expresses a fact). When we write $\varphi \in \mathcal{L}(\Phi)$, we mean that the fact φ is expressible in the language $\mathcal{L}(\Phi)$.

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 8 and 9 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 9 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. Consider the definition $\mathcal{P}^{fut}$, for example. Here the probability space we associate with a point (r, k) consists only of points (r', k) having the same global state as (r, k). The runs r' are the runs extending the global state $r(k)$. This means we can reason about the probability of a future event given a fixed global state. In contrast, a definition such as $\mathcal{P}^{post}$ allows for the possibility that the runs r' may extend any of a collection of global states, which may mean we no longer have the luxury of arguing about the probability of a future event given a fixed global state. At the very least, reasoning in terms of $\mathcal{P}^{fut}$ instead of $\mathcal{P}^{post}$ obviates the need for reasoning in terms of conditional probability. 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.

7 Probability assignments in asynchronous systems

We now turn our attention to choosing appropriate probability assignments in asynchronous systems. While we drop the synchronous assumption that $r_i(k) = r'_i(k')$ implies $k = k'$, we still assume the existence of a global clock, although agents in the systems may not have access to this clock or may only know approximate values of the clock. We remark that even in the context of asynchronous systems, the four sample space assignments discussed in the previous section—$\mathcal{S}^{post}$, $\mathcal{S}^{fut}$, $\mathcal{S}^j$, and $\mathcal{S}^{prior}$—still make perfect sense. The intuition motivating these definitions remains the same; in particular, Theorem 7 which says that $\mathcal{S}^j$ determines safe bets against p_j still holds.

A number of things *do* change, however. For one thing, Proposition 3 no longer holds, so many facts of interest become nonmeasurable. Equally important, Proposition 5, which says that consistent probability assignments further down in the lattice can all be obtained by conditioning on consistent probability assignments higher in the lattice, also fails in general. The reason it may fail is that if $\mathcal{S}' \leq \mathcal{S}$, we are no longer guaranteed that $S'_{i,c}$ is a *measurable* subset of $S_{i,c}$. For example, although $\mathcal{P}^j \leq \mathcal{P}^{post}$, $\mathit{Tree}^j_{i,c}$ need not be a measurable subset of $\mathit{Tree}_{i,c}$. If p_j can distinguish time 1 points from time 2 points but p_i cannot, and if c is a time 1 point, then $\mathit{Tree}^j_{i,c}$ consists only of time 1 points while $\mathit{Tree}_{i,c}$ consists of the time 1 and 2 points. In this case, $\mathit{Tree}^j_{i,c}$ is not a measurable subset of $\mathit{Tree}_{i,c}$, since $\mathit{Tree}^j_{i,c}$ does not contain all points of $\mathit{Tree}_{i,c}$ contained in runs passing through $\mathit{Tree}^j_{i,c}$. All our conditioning arguments used this measurability assumption. Consequently, it is no longer true that all consistent assignments can be obtained by conditioning on $\mathcal{P}^{post}$. For similar reasons, in general asynchronous systems, using $Tree_{i,c}$ and using $Tree^j_{i,c}$ in the definition of a safe bet does not necessarily give the same results. (The conditional probability argument used in the proof of Proposition 6 depends on the fact that the sets $Tree^j_{i,c}$ are measurable subsets of $Tree_{i,c}$.) The fact that Theorem 7 holds in asynchronous systems, saying that $\mathcal{S}^j$ still determines safe bets against p_j, depends on the fact that we use $Tree^j_{i,c}$ in the definition of a safe bet. We *can* prove analogues of Propositions 5 and 6 as well as Theorem 9, provided we assume that $S'_{i,c}$ is a measurable subset of $S_{i,c}$ for all agents p_i and points c.[15] Unfortunately, as we shall see, this measurability requirement does not hold in many cases of interest.

The situation is perhaps best illustrated by an example. Consider a simple asynchronous system (similar to the example in the introduction) in which agent p_3 tosses a fair coin 10 times — once each clock tick — and halts; agents p_1 and p_2 do nothing and never learn the outcome of the coin tosses. This system consists of a single computation tree, a complete binary tree of depth 10 with every transition labeled 1/2. Suppose agent p_1 does not have access to a clock, and so is unable to distinguish any of the global states in the tree. On the other hand, p_2 has access to the clock, and so can tell each time apart.

There are clearly 2^{10} possible runs in the system, one corresponding to each of the possible sequences of coin tosses. Since p_1 cannot distinguish any point on any of these runs, for every point c, the set $S^{post}_{1,c}$ consists of every point in the system. Which subsets of $S^{post}_{1,c}$ are measurable? Since the computation tree is finite, each individual run is a measurable set, so all sets of runs are measurable. And since the measurable subsets of $S^{post}_{1,c}$ are obtained by projecting measurable subsets of runs onto $S^{post}_{1,c}$, the sets in $\mathcal{X}^{post}_{1,c}$ are those consisting of all the points on some set of runs in the computation tree.

Let φ be the fact "the most recent coin toss landed heads", which is initially false at time 0. Although this is a fact about the global state, the set of points where it is true is not a measurable subset of $S^{post}_{1,c}$, since it does not consist of all the points on some subset of runs. This already shows that Proposition 3 fails in this case. Thus, we cannot talk about the probability that p_1 knows φ at a point c in the tree. We can talk about the inner and outer measure of $S^{post}_{1,c}(\varphi)$, however. Since the only nonempty measurable set contained in $S^{post}_{1,c}(\varphi)$ is the set of points on the single run in which the coin lands heads every time, the inner measure of this set is $1/2^{10}$; similarly, the outer measure is $1-(1/2^{10})$.

While values such as $1/2^{10}$ and $1-(1/2^{10})$ may seem somewhat strange at first glance, they are not totally unmotivated. Consider the situation of agent p_1 at a point c trying to figure out the probability of heads—the probability that φ is true—given only the probability on the runs. Agent p_1 has no idea which run it is in. The only run in which it is always the case that the most recent coin toss landed heads is the run where the coin lands heads on every toss: this run occurs with probability $1/2^{10}$. On the other hand, in all the runs except for the one in which the coin lands tails on every toss, it is possible that the most recent coin toss landed heads. Thus, in a set of runs of probability $1-(1/2^{10})$, it is possible that the most recent coin toss landed heads. This means that $1/2^{10}$ and $1-1/2^{10}$—the inner and outer measure of $S^{post}_{1,c}(\varphi)$—provide lower and upper bounds on the probability of being in a run where the most recent coin toss landed heads.

[15] In part (b) of this analogue of Theorem 9, we must also strengthen the definition of sufficiently rich to mean that for every global state there is a primitive proposition in Φ true at all points of all runs passing through this global state. This is due to the fact that consistent assignments in asynchronous systems allow a set $S_{i,c}$ to contain more than one point of a given run.

Now suppose that agent p_1 is betting against p_2. Since p_2 knows what the time is, each set $S^2_{1,(r,k)}$ consists of all the time k points. With respect to the sample space assignment $\mathcal{S}^2$, the fact φ is measurable. In fact, it is easy to see that $\mu^2(S_{1,c}(\varphi)) = 1/2$ for all points c. To sum up, we have $\mathcal{P}^{post}, c \models K_1^{[1/2^{10}, 1-(1/2^{10})]}\varphi$ and $\mathcal{P}^{post}, c \models \neg K_1^{1/2}\varphi$, while $\mathcal{P}^2, c \models K_1^{1/2}\varphi$.[16]

This may seem somewhat counterintuitive, since it seems to suggest that p_1 must play more conservatively against a copy of itself than against p_2, who knows more. This is especially so since there is another line of reasoning about this situation which would lead p_1 to conclude that it knows that the probability that the most recent coin toss landed heads is $1/2$, even without considering p_2. Agent p_1 reasons as follows: "The current time is k, although I do not know what k is. Regardless of the particular value of k, the probability that the k^{th} coin toss lands heads is $1/2$, and hence I know the most recent coin toss landed heads with probability $1/2$." The sample space assignment that captures this intuition would associate with the point (r, k) and agent p_1 the set of *time k points* in (r, k)'s computation tree agent p_1 considers possible at (r, k) (as opposed to considering *all* the points in the computation tree that p_1 considers possible, as is done by $\mathcal{P}^{post}$). But this is precisely the assignment $\mathcal{S}^2$!

In order to understand this situation a little better, let us reconsider the assignment $\mathcal{P}^{post}$. We claim that the reason the interval $[1/2^{10}, (1 - 1/2^{10})]$ arises here is different from the reason intervals arise in the context of the synchronous systems studied in the preceding section. In the context of synchronous systems, because p_j's strategy depends on its local state and p_i does not know which local state p_j is currently in, p_i has to partition $\mathcal{K}_i(c)$ and view each element of the partition as an independent probability space, computing the probability of φ separately in each one. A formula such as $K_i^{[\alpha,\beta]}\varphi$ holds when the probability of φ can range from α to β in the different probability spaces. In our current example, however, there is only one probability space; the interval arises because of the nonmeasurability of φ. Depending on how "lucky" p_1 is in the choice of where in each run it tests for heads, the probability of getting heads could range from $1/2^{10}$ to $1 - (1/2^{10})$.

We can view the nonmeasurability that arises due to asynchrony as a new element of uncertainty that an adversary can exploit. Intuitively, in the coin tossing example, when p_1 plays against (a copy of) itself, since p_1 does not know where in the run it is, an adversary gets to choose that. On the other hand, when playing against p_2, at least p_1 knows that all the worlds in a given sample space are time k points, for some fixed k. We can view our analysis where we obtain the answer $1/2$ without invoking p_2 as implicitly assuming an adversary who chooses the time k the test for φ is to be performed. Such an adversary is an adversary of the third type mentioned in the introduction. Given any time k chosen by this adversary, the probability of φ is $1/2$.

We can formalize this analysis as follows. With each time k we associate a separate computation tree corresponding to the adversary A_k that chooses time k to test for the truth of a fact, in our case for the truth of φ. The probability space for p_1 at each point in the tree corresponding to A_k consists of the time k points in the tree, each of which is assigned equal probability. In each of these probability spaces the probability of heads is $1/2$, so p_1 knows that the most recent coin toss landed heads with probability $1/2$.

There is no reason, however, to restrict this third type of adversary to simply making an

[16] Note that this does not contradict Theorem 9, since Theorem 9 would hold only if $S^2_{i,c}$ is a measurable subset of $S_{i,c}$ for all p_i and c, which we have already noted is not the case.

initial choice of the stopping time. Suppose we have fixed a collection of adversaries of the first type (the computation trees) and an adversary of the second type (say p_j). We define a *cut* through $\mathit{Tree}^j_{i,c}$ to be a subset of $\mathit{Tree}^j_{i,c}$ containing precisely one point from every run passing through $\mathit{Tree}^j_{i,c}$: every run passing through $\mathit{Tree}^j_{i,c}$ is cut precisely once by such a set of points. We define a type three adversary to be a function mapping an agent p_i and a point c to a cut through $\mathit{Tree}^j_{i,c}$. Intuitively, p_i and p_j are betting on a fact φ, but neither knows precisely where in the run the bet is taking place; it is the third type of adversary who determines where in the run the bet is actually made. The cut through $\mathit{Tree}^j_{i,c}$ chosen by the adversary is the set of points at which the adversary will cause the bet to take place when the local states of p_i and p_j are given by c.

In the example above, when p_1 plays against a copy of itself, the adversary chooses one cut per computation tree, since p_1 considers all points in the computation tree possible. In the case of p_1 playing against p_2 (who knows the time), the adversary chooses one cut for every time k; this cut must in fact consist of all time k points in the tree. (In general, if we are considering a set of time k points, the only allowable cut is the one consisting of all points. This is why the issue of an adversary choosing such cuts does not arise when considering synchronous systems.)

To make formal sense of this, suppose we are given a set $\mathcal{A}$ of type one adversaries (determining the possible initial nondeterministic choices). This determines a set of computation trees, as we have already discussed. Fix a type two adversary, say p_j. Let $\mathcal{C}$ be a set of type three adversaries in this collection of computation trees (so that the adversaries in $\mathcal{C}$ choose stopping times). Notice that the definition of $\mathcal{C}$ depends on $\mathcal{A}$ and p_j. We can then construct one computation tree $\mathcal{T}_{A,C}$ for each $A \in \mathcal{A}$ and $C \in \mathcal{C}$. For a fixed $A \in \mathcal{A}$, the computation trees $\mathcal{T}_{A,C}$ look identical (essentially just like $\mathcal{T}_A$) for all choices of $C \in \mathcal{C}$ except that we put C into the environment state at each point in $\mathcal{T}_{A,C}$. The sample space assignment $\mathcal{S}^{\mathcal{C}}$ maps an agent p_i and a point c of a tree $\mathcal{T}_{A,C}$ to a sample space $S^{\mathcal{C}}_{i,c} \subseteq \mathit{Tree}^j_{i,c}$ such that for each run $r \in \mathcal{R}(\mathit{Tree}^j_{i,c})$, exactly one point $(r,k) \in \mathit{Tree}^j_{i,c}$ is in $S^{\mathcal{C}}_{i,c}$. Intuitively, this is the point in r where the test is performed. Note that if we consider two adversaries $C, C' \in \mathcal{C}$ and two corresponding points c and c' in $\mathcal{T}_{A,C}$ and $\mathcal{T}_{A,C'}$, the sample spaces $S_{i,c}$ and $S_{i,c'}$ used by p_i at these two points will in general be different: at c, it is C that determines at which point in each run in the tree that p_i considers possible at c the test will be performed, while at c' it is C' that makes this determination. Notice that, in the presence of this third type of adversary, it is no longer the case that all sample space assignments defined in asynchronous systems are standard assignments. For example, it no longer need be the case that $c \in S^{\mathcal{C}}_{i,c}$.

Intuitively, playing against a copy of yourself places no constraints on this third type of adversary. To make this precise, once we fix a set of adversaries of the first type $\mathcal{A}$ (and a copy of yourself as the type 2 adversary) and consider the resulting system, we can take $\mathit{pts}(\mathcal{A})$ to be the set of all possible adversaries of the third type in this system. Let $\mathcal{S}^{pts}$ be the sample space assignment $\mathcal{S}^{\mathcal{C}}$ defined above where $\mathcal{C} = \mathit{pts}(\mathcal{A})$, and let $\mathcal{P}^{pts}$ be the induced probability assignment.

Proposition 10: $\mathcal{P}^{post}, c \models K_i^{[\alpha,\beta]}\varphi$ iff $\mathcal{P}^{pts}, c \models K_i^{[\alpha,\beta]}\varphi$, for every fact φ, agent p_i, and point c.

The proof of this result shows that $\mathcal{P}^{post}$ can be understood in asynchronous systems in terms of an adversary that chooses as the time for the test to be performed the worst possible time from p_i's point of view.[17]

Of course, there is no reason to assume that a type three adversary must either be restricted to choosing horizontal cuts of time k points or be allowed to choose completely arbitrary cuts of points. Other intermediate definitions seem plausible as well. One can imagine a partially synchronous model in which processors cannot tell time but are guaranteed that, for every k, all processors take their k^{th} step within some time interval of width δ. It would seem reasonable to require the adversary of the third type, rather than selecting horizontal time k cuts or totally arbitrary cuts, to select cuts with the property that every point in the cut is a time k point for some k falling in some interval of width δ. We can also generalize the notion of type three adversary slightly so as not to require that it choose a cut, but rather have it choose at most one point per run. The intuition here is that this adversary simply does not give p_i the chance to bet in certain runs. In our coin tossing example, such an adversary could allow p_i to bet on heads only when the coin has landed tails. The issue of defining reasonable adversaries of the third type deserves further study.

We close this section with a comparison of our definition of probability in asynchronous systems with that of [FZ88a]. The probability assignment used in [FZ88a] in the asynchronous setting has much the same flavor as that of our $\mathcal{P}^{pts}$. Rather than assuming that the adversary chooses at a point c a cut of *points* through $Tree_{i,c}$, however, Fischer and Zuck assume that the adversary chooses a cut of *global states* through $Tree_{i,c}$; that is, a set of global states appearing in $Tree_{i,c}$ with the property that no two global states lie on the the same run. Intuitively, this means that if the adversary performs the test at one point, it performs the test at all other points with the same global state. This seems like a reasonable restriction, but it leads to some unexpected consequences.

Let us call the class of adversaries considered in [FZ88a] *state*, let $\mathcal{S}^{state}$ be the sample space assignment $\mathcal{S}^{\mathcal{C}}$ defined above where $\mathcal{C} = state$, and let $\mathcal{P}^{state}$ be the induced probability assignment. Rather than giving formal definitions here, we give an example to show how $\mathcal{P}^{state}$ differs from $\mathcal{P}^{pts}$. Consider a system in which p_1 tosses a biased coin which lands heads with probability .99 and tails with probability .01. The system consists of two runs we can denote by h and t and four points corresponding to times 0 and 1 in runs h and t. The computation tree has only three nodes, a root R encoding the points $(h,0)$ and $(t,0)$, a node H corresponding to the point $(h,1)$, and a node T corresponding to $(t,1)$. Suppose p_2 is able to distinguish only the point $(h,1)$ from the remaining three points and suppose that φ is the fact "the coin lands heads" (so that φ is true at $(h,0)$ and $(h,1)$, and false elsewhere). Let c be a time 0 point, say $(t,0)$, and consider the probability with which p_2 knows φ with respect to $\mathcal{P}^{pts}$ and $\mathcal{P}^{state}$. An adversary in *pts* can either choose $\{(h,0),(t,0)\}$ or $\{(h,0),(t,1)\}$ as the set of points to perform the experiment; φ is true with probability .99 with respect to both sets. It follows that $\mathcal{P}^{pts}, c \models K_2^{.99}\varphi$; in fact we have $\mathcal{P}^{pts}, c \models K_2^{[.99,.99]}\varphi$. Similarly, an adversary in *state* can choose either the node R or the node T as a state at which to perform the experiment, since these are the cuts of global states contained in $\{R,T\}$. The choice of R corresponds to the adversary in *pts* that chooses $\{(h,0),(t,0)\}$. However, the choice of T does not correspond to $\{(h,0),(t,1)\}$. In fact, there is no adversary in *state* corresponding to this adversary in *pts*,

[17]Another interpretation of this result is that the language obtained by closing a set of formulas under the standard boolean connectives and the modal operators K_i^α cannot distinguish the assignments $\mathcal{P}^{post}$ and $\mathcal{P}^{pts}$. We note that the richer language of [FH94] can distinguish these assignments.

since it would amount to choosing the nodes R and T, both of which lie on the same run. With respect to the choice R, φ holds with probability .99; with respect to the choice T, φ holds with probability 0. Thus, we get $\mathcal{P}^{state}, c \models K_2^{[0,.99]}\varphi$. In some sense it seems that $\mathcal{P}^{pts}$ is giving the more reasonable answer here. Since p_2 knows that, *a priori*, the coin will land heads with high probability, and its information has not eliminated either run, it should still consider heads extremely probable.[18]

8 An application: coordinated attack

As an application of the framework we have developed, we specify the probabilistic coordinated attack problem in terms of knowledge and probability, and show how the nature of the problem changes as the probability assignment changes. Recall from Section 4 that deterministic coordinated attack requires that two generals A and B coordinate a synchronized attack on a common enemy, in spite of the fact that A and B can communicate only by sending messages via messengers who can be captured by the enemy. If we define φ_{CA} to be the fact "A attacks iff B attacks," then we can take the problem's correctness condition to be that φ_{CA} holds at all points. The analogous correctness condition for probabilistic coordinated attack would be that φ_{CA} holds with high probability. The meaning of this slippery phrase "with high probability" is the topic of this section.

Halpern and Moses [HM90] have already demonstrated a strong relationship between *deterministic* coordinated attack and a state of knowledge called *common knowledge*. Intuitively, a formula φ is common knowledge if all agents know φ, all agents know all agents know φ, and so on *ad infinitum*. Formally, given a set $G \subseteq \{p_1, \ldots, p_n\}$ of agents, it is customary to define *everyone in G knows φ*, denoted $E_G\varphi$, by

$$E_G\varphi \equiv \bigwedge_{p_i \in G} K_i\varphi,$$

and to define *φ is common knowledge to G*, denoted $C_G\varphi$, by

$$C_G\varphi \equiv E_G\varphi \wedge E_G E_G\varphi \wedge \cdots \wedge E_G^m\varphi \wedge \cdots.$$

It is not hard to prove that common knowledge satisfies the following statements [HM92]:

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 $X \equiv E_G(\varphi \wedge X)$. 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 [HM92]. In fact, it is common to define common knowledge as the greatest fixed point of this equation, since this is the stronger definition in general and is the definition that extends most easily to other settings [HM90, FH94, PT92].

[18] Note that this example also shows that the adversaries in *state* are examples of the more general adversaries discussed above, that do not necessarily choose one point per run. For example, the adversary choosing the global state T does not choose a point in the run h.

The induction rule—in the simple case that $\psi \equiv \varphi$—says that if φ is a "public fact" in the sense that everyone knows φ whenever φ is true, then it is common knowledge whenever it is true.

What Halpern and Moses show is that common knowledge of a certain fact is a necessary condition for coordinated attack. They also show that common knowledge of nontrivial facts cannot be attained in systems such as systems in which messages may fail to be delivered, and hence that coordinated attack is not possible in systems such as ours (cf. [Gra78]). Given this relationship between deterministic attack and common knowledge, it is natural to ask about a relationship between probabilistic attack and probabilistic common knowledge.

By direct analogy to the fixed-point definition of common knowledge, Fagin and Halpern [FH94] define *probabilistic common knowledge* of φ to be the greatest fixed point of the $X \equiv E_G^\alpha(\varphi \wedge X)$, where $E_G^\alpha \varphi \equiv \bigwedge_{p_i \in G} K_i^\alpha \varphi$. This definition is not equal to the infinite conjunction of the formulas $(E_G^\alpha)^k \varphi$ for $k > 0$, but it is easy to show that this definition satisfies the obvious analogues of the fixed point axiom and induction rule given above.

Returning to deterministic attack, notice that since φ_{CA} is true at all points, the induction rule implies $C_G \varphi_{CA}$ holds at all points. In fact, since $C_G \varphi_{CA}$ implies φ_{CA}, an alternative specification of coordinated attack is that $C_G \varphi_{CA}$ holds at all points (and not that φ_{CA} holds at all points). Suppose we fix some $\alpha > 0$ and take the specification of probabilistic attack to be that $C_G^\alpha \varphi_{CA}$ holds at all points. Are there implementations of this version of probabilistic attack? The answer depends on the choice of probability assignment. Stronger assignments yield stronger notions of probabilistic common knowledge, and hence make stronger requirements of the implementation.

Consider the assignment $\mathcal{P}^{fut}$. Here the opponent 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 later point of any run extending this point can satisfy φ_{CA}. At this point φ_{CA} holds with probability 0 (according to $\mathcal{P}^{fut}$), so it easily follows that $C_G^\alpha \varphi_{CA}$ cannot hold at all points. 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 opponent.

Next consider the assignment $\mathcal{P}^{post}$. Here the opponent 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_{CA}, the agent knows φ_{CA} does not hold, and hence neither does $C_G^\alpha \varphi_{CA}$. Consequently, our first implementation CA_1 of the probabilistic attack problem does not have the property that $C_G^\alpha \varphi_{CA}$ holds at all points (with respect to $\mathcal{P}^{post}$), but our second implementation CA_2 does. This can be proved by first observing that $E_G^\alpha \varphi_{CA}$ 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 $C_G^\alpha \varphi_{CA}$.

Notice that with respect to any consistent probability assignment, if at some point an agent in G knows φ_{CA} does not hold, then $C_G^\alpha \varphi_{CA}$ cannot hold at this point (since $C_G^\alpha \varphi_{CA}$ implies $E_G^\alpha \varphi_{CA}$ by the fixed point axiom, while $K_i \neg\varphi$ implies $\neg E_G^\alpha \varphi_{CA}$ for all $i \in G$). Consequently, it cannot be the case that $C_G^\alpha \varphi_{CA}$ holds at all points of CA_1 with respect to any consistent assignment. Is it possible for $C_G^\alpha \varphi_{CA}$ to hold at all points of CA_1 with respect to *any* probability assignment? Since this algorithm guarantees φ_{CA} 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_{CA} with probability α at all points of the system. Since $E_G^\alpha \varphi_{CA}$ holds at all points, it follows by the induction rule that $C_G^\alpha \varphi_{CA}$ holds at all points as well.

We summarize our discussion in the following proposition.

Proposition 11:

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

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

3. A protocol achieves probabilistic coordinated attack with respect to $\mathcal{P}^{fut}$ iff it achieves coordinated attack. Hence no such protocol exists in which the generals actually attack, but do not attack in the absence of messages.

This proposition shows how increasing the power of the opponent (moving down in the lattice) strengthens the kind of guarantees that can be made for probabilistic attack. As we have already seen, $\mathcal{P}^{prior}$ corresponds to probability over the runs. And indeed, CA_1 does give us coordination in almost all the runs. However, as we observed in Section 4, by following protocol CA_1, general A can attack even though A is certain that the attack will not be coordinated. Since CA_2 achieves coordinated attack with respect to $\mathcal{P}^{post}$, this situation cannot arise with CA_2. Finally, part 3 of Proposition 11 shows that achieving coordinated attack with high probability with respect to the knowledge of an agent that knows everything that has happened in the past is too much to expect in this setting.

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; i.e. if c is a time 0 point in $\mathcal{T}_A$ and $\mathcal{R}_A(\varphi)$ is the set of runs in $\mathcal{T}_A$ satisfying a fact φ about the run, then

$$\mu_A(\mathcal{R}(\varphi)) = \mu_{i,c}^{post}(\mathit{Tree}_{i,c}(\varphi)) = \mu_{i,c}^{j}(\mathit{Tree}_{i,c}^{j}(\varphi)) = \mu_{i,c}^{fut}(\mathit{Pref}_{i,c}(\varphi)) = \mu_{i,c}^{prior}(\mathit{All}_{i,c}(\varphi)).$$

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 stronger definition of probabilistic coordinated attack is considered in [FZ88a]: it is required only that the conditional probability that both parties attack together, given that one of the parties attacks, is at least α.[19] It is then shown in [FZ88a] that this form

[19] Although it is not clear from the definition of probabilistic attack given in [FZ88a] over what the probability is being taken, the results given clearly assume that the probability is being taken over the runs.

of probabilistic coordinated attack corresponds to all the agents having *average belief* of α that the attack will be coordinated. We can reinterpret their 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 $\mathcal{P}^{prior}$. What is interesting here is that the probability space used by [FZ88a] for this analysis is not $\mathcal{P}^{post}$, but an inconsistent probability assignment. This point is not significant in their example, since their results are consistent with one's intuition. In general, however, it seems unreasonable to use inconsistent probability assignments, since one can be led to counterintuitive results with such assignments. 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 $K_A^\alpha \varphi_{CA}$ and $K_A \neg\varphi_{CA}$ 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 computer science applications consider only probabilities over runs (such applications typically require only that a condition P hold throughout a large fraction of the runs, 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 when the prover knows perfectly well that it has already leaked information to the verifier, and may continue to do so. 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 the future.

9 Conclusion

We have provided a framework for reasoning about knowledge and probability in distributed systems. We have illustrated the fact that no single definition of probabilistic knowledge is appropriate in all contexts, and we have shown that each definition of probabilistic knowledge is best understood in terms of a particular choice of three types of adversaries. The primary technical contribution of this work has been to show how to construct the most appropriate definition in the context of any particular choice of these adversaries. Introducing the notion of an adversary has helped 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.

The challenge that remains is to apply this new-found understanding to real problems. In order to apply our results, there is one extension of this work that might be useful. We have shown how to construct the most appropriate probability space for a given opponent in a betting game, but we have made no assumptions about the strategy the adversary is following. One potentially fruitful line of research is to understand how our results are effected if we make assumptions about the strategies the adversary p_j is allowed to follow, such as assuming that p_j is trying to maximize its payoff and not simply trying to break even. If p_j is willing to take risks to increase its payoff, then this might decrease the minimum payoff p_i is willing to accept on a bet. Such extensions could be useful when applying this framework to applications from game theory and economics, such as contract negotiation.

The area of application that appears to have the most potential is the analysis of probabilistic distributed protocols, especially cryptographic protocols. The most promising direction could be the specification of cryptographic protocols in terms knowledge and probability. Correctness conditions in cryptography are typically described at a very low-level of abstraction, involving Turing machines and complicated statements of conditional probability. If it were possible to specify these protocols at a higher level of abstraction in terms of knowledge and probability, then it could be possible to reason about these protocols at a higher level of abstraction using the axioms and inference rules for probabilistic knowledge given by Fagin and Halpern [FH94]. Some progress in this direction has already been made. Fischer and Zuck [FZ87] have used notions of knowledge, probability and complexity theory to analyze a particular interactive proof for quadratic residuosity, and Halpern, Moses, and Tuttle [HMT88] have used related notions to completely characterize interactive and zero knowledge proof systems in terms of knowledge. The current difficulty in continuing down this path is that we do not have helpful axioms and inference rules for reasoning about complexity theory, and hence it is difficult to use these characterizations to reason about these protocols in a mechanical way. On the other hand, reasoning about the purely probabilistic aspects of these protocols is certainly possible, using the axioms and inference rules of [FH94].

A Proofs of results

This appendix contains the proofs of all results claimed in the paper.

Proposition 1: If $S_{i,c}$ is state generated and satisfies REQ_1, then $S_{i,c}$ satisfies REQ_2.

Proof: Given a global state g, let G_g be the set of points (r,k) with $r(k) = g$, and let $\mathcal{R}_g$ be the set of runs through g. By our technical assumption that the global state encodes the adversary and the past history of the run, each global state is contained in precisely one computation tree, and appears precisely once in this tree. Thus, G_g and $\mathcal{R}_g$ are contained in a single computation tree, and $\mathcal{R}_g = \mathcal{R}(G_g)$. Since $S_{i,c}$ is state generated, $S_{i,c}$ is the union of a collection of sets of the form G_g. Since $S_{i,c}$ satisfies REQ_1, it is contained in a single computation tree $\mathcal{T}_A = (\mathcal{R}_A, \mathcal{X}_A, \mu_A)$; and since a single computation tree contains at most a countable number of global states, $S_{i,c}$ is a countable union of sets of the form G_g. Thus, $\mathcal{R}(S_{i,c})$ is the countable union of sets of the form $\mathcal{R}_g = \mathcal{R}(G_g)$ with g a global state in $\mathcal{T}_A$. By the definition of $\mathcal{T}_A$, each set $\mathcal{R}_g$ is a measurable set of runs with positive measure, and hence their countable union $\mathcal{R}(S_{i,c})$ must also be a measurable set with positive measure. It follows that $S_{i,c}$ satisfies REQ_2. $\square$

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

Proof: We must show (see [Hal50]) that $\mathcal{X}_{i,c}$ is a set of subsets of $S_{i,c}$ including $S_{i,c}$ that is closed under the formation of complements and countable unions, and that $\mu_{i,c}$ is a nonnegative, countably additive function on $\mathcal{X}_{i,c}$ satisfying $\mu_{i,c}(\emptyset) = 0$.

Let $\mathcal{T}(c) = (\mathcal{R}_A, \mathcal{X}_A, \mu_A)$. Since $S_{i,c} = Proj(\mathcal{R}_A, S_{i,c})$ and $\mathcal{R}_A \in \mathcal{X}_A$, we have $S_{i,c} \in \mathcal{X}_{i,c}$. If $X \in \mathcal{X}_{i,c}$, then $X = Proj(R, S_{i,c})$ for some $R \in \mathcal{X}_A$; since $\mathcal{X}_A$ is closed under complementation, $R^c \in \mathcal{X}_A$ and $X^c = Proj(R^c, S_{i,c}) \in \mathcal{X}_{i,c}$, and hence $\mathcal{X}_{i,c}$ is closed under complementation. If $X_1, X_2, \ldots$ is a countable collection of sets from $\mathcal{X}_{i,c}$, then for each j we have $X_j = Proj(R_j, S_{i,c})$ for some $R_j \in \mathcal{X}_A$. Since $\mathcal{X}_A$ is closed under countable union, $R = \cup_j R_j \in \mathcal{X}_A$. It follows that

$$X = \cup_j X_j = \cup_j Proj(R_j, S_{i,c}) = Proj(\cup_j R_j, S_{i,c}) = Proj(R, S_{i,c}),$$

so $X \in \mathcal{X}_{i,c}$ and $\mathcal{X}_{i,c}$ is closed under countable union.

Since $S_{i,c}$ is contained in a single computation tree by REQ_1, and since $\mathcal{R}(S_{i,c}) \in \mathcal{X}_A$ and $\mu_A(\mathcal{R}(S_{i,c})) > 0$ by REQ_2, conditional probability with respect to $\mathcal{R}(S_{i,c})$ is well-defined, and hence $\mu_{i,c}$ is well-defined. Clearly, $\mu_{i,c}$ is nonnegative since μ_A is. Furthermore, $\mu_{i,c}(\emptyset) = \mu_A(\emptyset)/\mu_A(\mathcal{R}(S_{i,c})) = 0$. Finally, suppose $X_1, X_2, \ldots$ is a countable collection of pairwise-disjoint sets in $\mathcal{X}_{i,c}$. For each j, we have $X_j = Proj(R_j, S_{i,c})$ for some $R_j \in \mathcal{X}_A$. We can assume every run in R_j passes through $S_{i,c}$, or we can replace R_j with the measurable set $\mathcal{R}(S_{i,c}) \cap R_j$; and we can assume the R_j are pairwise disjoint, since if r is contained in both R_j and R_k then some point on r is contained in both $X_j = Proj(R_j, S_{i,c})$ and $X_k = Proj(R_k, S_{i,c})$, contradicting the pairwise-disjointness of X_j and X_k. It follows from the pairwise-disjointness of the $R_j = \mathcal{R}(X_j)$ that

$$\mu_{i,c}(\cup_j X_j) = \frac{\mu_A(\mathcal{R}(\cup_j X_j))}{\mu_A(\mathcal{R}(S_{i,c}))} = \frac{\mu_A(\cup_j \mathcal{R}(X_j))}{\mu_A(\mathcal{R}(S_{i,c}))} = \sum_j \frac{\mu_A(\mathcal{R}(X_j))}{\mu_A(\mathcal{R}(S_{i,c}))} = \sum_j \mu_{i,c}(X_j),$$

and hence $\mu_{i,c}$ is countably additive. $\square$

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

Proof: Recall that $\mathcal{L}(\Phi)$ is state generated if all the primitive propositions in Φ are facts about the global state. Recall also that φ is measurable with respect to $\mathcal{S}$ if $S_{i,c}(\varphi) \in \mathcal{X}_{i,c}$ for all agents p_i and points c. Fix an agent p_i and a point c. Let S_k denote the set of time k points in the computation tree $\mathcal{T}(c)$ containing c. We claim it is enough to show that

$(*)$ $\quad \mathcal{R}(S_k(\varphi))$ is a measurable set of runs for all times k and all formulas $\varphi \in \mathcal{L}(\Phi)$.

To see this, notice that since $\mathcal{S}$ is a consistent assignment in a synchronous system, $S_{i,c}$ contains only time k points for some k. In fact, since $S_{i,c}$ satisfies REQ_1, all points of $S_{i,c}$ are contained in $\mathcal{T}(c)$. Consequently, $\mathcal{R}(S_{i,c}(\varphi)) = \mathcal{R}(S_{i,c}) \cap \mathcal{R}(S_k(\varphi))$. Since $\mathcal{R}(S_{i,c})$ is measurable by REQ_2, condition $(*)$ will imply $\mathcal{R}(S_{i,c}(\varphi))$ is measurable. It will follow that $S_{i,c}(\varphi)$ is a measurable subset of $S_{i,c}$.

The proof of $(*)$ proceeds by induction on the structure of φ. If φ is a primitive proposition in Φ, then since $\mathcal{L}(\Phi)$ is state generated we know that φ must be a fact about the global state. Arguments similar to those used for Proposition 1, therefore, suffice to show that $\mathcal{R}(S_k(\varphi))$ is a measurable set of runs. The cases of negation and conjunction follow immediately from the fact that measurable sets are closed under complementation and intersection. For $\varphi = K_i\psi$, since $K_i\psi$ is a fact about the global state, the arguments for such a formula is identical to the argument for primitive propositions above.

For $\varphi = Pr_i(\psi) \geq \alpha$, consider any time k point d. Since $\mathcal{S}$ is inclusive, we know that $d \in S_{i,d}$. Since $\mathcal{S}$ is a consistent assignment in a synchronous system, we know that $S_{i,d}$ contains only time k points from $\mathcal{T}(d)$. It follows that S_k is the union of sets of time k points of the form $S_{i,d}$. Moreover, since $\mathcal{S}$ is uniform, the $S_{i,d}$ actually partition S_k. Finally, since each $S_{i,d}$ is state generated and since there are at most a countable number of time k global states in any given tree, we see that S_k is partitioned into a countable collection of sets of the form $S_{i,d}$. Now, since we consider only uniform sample space assignments, it is easy to check that φ is true at either all or none of the points in $S_{i,d}$. It follows that $S_k(\varphi)$ is partitioned into a countable collection of sets of the form $S_{i,d}$, and hence that $\mathcal{R}(S_k(\varphi))$ is partitioned into a countable collection of sets of the form $\mathcal{R}(S_{i,d})$. Since the sets $\mathcal{R}(S_{i,d})$ are measurable by REQ_2, so is their countable union $\mathcal{R}(S_k(\varphi))$.

For $\varphi = \bigcirc\psi$, notice that ψ is true at $(r, k+1)$ iff $\bigcirc\psi$ is true at (r,k). It follows that $\mathcal{R}(S_k(\bigcirc\psi)) = \mathcal{R}(S_{k+1}(\psi))$, and hence by the inductive hypothesis for ψ that $\mathcal{R}(S_k(\varphi)) = \mathcal{R}(S_k(\bigcirc\psi))$ is a measurable set of runs. In fact, a simple extension of this argument (by induction on ℓ) shows that if $\mathcal{R}(S_k(\psi))$ is measurable then so is $\mathcal{R}(S_k(\bigcirc^\ell\psi))$.

For $\varphi = \psi\, U\, \theta$, define $\psi\, U_0\, \theta$ to be the formula θ, and define $\psi\, U_\ell\, \theta$ for $\ell > 0$ to be the formula $\psi \wedge \ldots \wedge \bigcirc^{\ell-1}\psi \wedge \bigcirc^\ell\theta$. It is easy to see that $\psi\, U\, \theta$ is true at a point d iff $\psi\, U_\ell\, \theta$ is true at d for some $\ell \geq 0$. Thus, $S_k(\psi\, U\, \theta) = \bigcup_{\ell\geq 0} S_k(\psi\, U_\ell\, \theta)$ and hence $\mathcal{R}(S_k(\psi\, U\, \theta)) = \bigcup_{\ell \geq 0}\mathcal{R}(S_k(\psi\, U_\ell\, \theta))$. Since the induction hypothesis holds for the subformulas ψ and θ, the preceding paragraph shows that each set $\mathcal{R}(S_k(\psi\, U_\ell\, \theta))$ is also measurable, and hence so is their countable union $\mathcal{R}(S_k(\varphi)) = \mathcal{R}(S_k(\psi\, U\, \theta))$. $\square$

Proposition 4: If $\mathcal{S}$ and $\mathcal{S}'$ are standard assignments satisfying $\mathcal{S}' \leq \mathcal{S}$, then for every agent p_i and point c, the set $S_{i,c}$ can be partitioned into sets of the form $S'_{i,d}$ with $d \in S_{i,c}$.

Proof: Suppose that $\mathcal{S}$ and $\mathcal{S}'$ are standard assignments satisfying $\mathcal{S}' \leq \mathcal{S}$. Since $\mathcal{S}'$ is inclusive and $\mathcal{S}$ is uniform, we have $d \in S'_{i,d} \subseteq S_{i,d} = S_{i,c}$ for every $d \in S_{i,c}$, and hence $S_{i,c}$ is the union of the $S'_{i,d}$ with $d \in S_{i,c}$. Furthermore, since $\mathcal{S}'$ is uniform, two sets $S'_{i,d}$ and $S'_{i,e}$ are either equal or disjoint, and hence $S_{i,c}$ can be partitioned into sets of the form $S'_{i,d}$ with $d \in S_{i,c}$. $\square$

Proposition 5: In a synchronous system, if $\mathcal{P}'$ and $\mathcal{P}$ are consistent, standard assignments satisfying $\mathcal{P}' \leq \mathcal{P}$, then for all agents p_i, all points c, and all measurable subsets $S' \in \mathcal{X}'_{i,c}$

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

Proof: Fix an agent p_i, a time k point c of $\mathcal{T}_A = (\mathcal{R}_A, \mathcal{X}_A, \mu_A)$, and a set $S' \in \mathcal{X}'_{i,c}$.

(a) Since $S' \in \mathcal{X}'_{i,c}$, there must exist some subset $\mathcal{R}' \in \mathcal{X}_A$ such that $S' = Proj(\mathcal{R}', S'_{i,c})$. Without loss of generality, we can assume that $\mathcal{R}' \subseteq \mathcal{R}(S'_{i,c})$ (since we can replace $\mathcal{R}'$ with $\mathcal{R}' \cap \mathcal{R}(S'_{i,c})$, which must also be measurable since REQ_2 guarantees $\mathcal{R}(S'_{i,c})$ is measurable). Since $S'_{i,c} \subseteq S_{i,c}$ and both $S'_{i,c}$ and $S_{i,c}$ consist of time k points (since $\mathcal{P}$ and $\mathcal{P}'$ are consistent assignments in a synchronous system), we have

$$Proj(\mathcal{R}', S'_{i,c}) = \{(r', k) \in S'_{i,c} : r' \in \mathcal{R}'\} = \{(r', k) \in S_{i,c} : r' \in \mathcal{R}'\} = Proj(\mathcal{R}', S_{i,c}).$$

Thus $S' = Proj(\mathcal{R}', S_{i,c})$, which shows that S' is a measurable subset of $S_{i,c}$.

(b) By part (a), it follows that $S'_{i,c}$ is a measurable subset of $S_{i,c}$. Since we have restricted to standard assignments $\mathcal{S}'$, we know that $S'_{i,c}$ is state generated, and arguments similar to the proof of Proposition 1 show that $\mu_{i,c}(S'_{i,c}) > 0$.

(c) Tracing through definitions, we see

$$\mu'_{i,c}(S') = \frac{\mu_A(\mathcal{R}(S'))}{\mu_A(\mathcal{R}(S'_{i,c}))} = \frac{\mu_A(\mathcal{R}(S'))/\mu_A(\mathcal{R}(S_{i,c}))}{\mu_A(\mathcal{R}(S'_{i,c}))/\mu_A(\mathcal{R}(S_{i,c}))} = \frac{\mu_{i,c}(S')}{\mu_{i,c}(S'_{i,c})} = \mu_{i,c}(S'|S'_{i,c}).$$

□

Proposition 6: In a synchronous system, for all facts φ, all agents p_i and p_j, and all points c, the rule $Bet(\varphi, \alpha)$ is *Tree*-safe for p_i at c iff $Bet(\varphi, \alpha)$ is $Tree^j$-safe for p_i at c.

Proof: Since $\mathcal{P}^j$ and $\mathcal{P}^{post}$ are standard probability assignments satisfying $\mathcal{P}^j \leq \mathcal{P}^{post}$, the sample space $Tree_{i,c}$ can be partitioned into the sample spaces $Tree^j_{i,d}$ with $d \in Tree_{i,c}$, and each such $Tree^j_{i,d}$ is a measurable subset of $Tree_{i,c}$ by Proposition 5. The law of conditional expectation, therefore, states that

$$E_{Tree_{i,c}}[W_f] = \sum E_{Tree_{i,c}}[W_f | Tree^j_{i,d}]\, \mu_{i,c}(Tree^j_{i,d}),$$

where the summation is taken over the sets of the form $Tree^j_{i,d}$ partitioning in $Tree_{i,c}$. Since $\mathcal{P}^j \leq \mathcal{P}^{post}$, we can use part (c) of Proposition 5 to prove that $E_{Tree_{i,c}}[W_f | Tree^j_{i,d}] = E_{Tree^j_{i,d}}[W_f]$, and hence that

$$E_{Tree_{i,c}}[W_f] = \sum E_{Tree^j_{i,d}}[W_f]\, \mu_{i,c}(Tree^j_{i,d}).$$

Suppose $Bet(\alpha, \varphi)$ is $Tree^j$-safe for p_i at c. Then $E_{Tree^j_{i,d}}[W_f] \geq 0$ for all f and for all points d agent p_i considers possible at c, which implies $E_{Tree_{i,e}}[W_f] \geq 0$ for all f and for all points e agent p_i considers possible at c, and hence that $Bet(\alpha, \varphi)$ is *Tree*-safe for p_i at c.

Conversely, suppose $Bet(\alpha, \varphi)$ is *not* $Tree^j$-safe for p_i at c. Then $E_{Tree^j_{i,d}}[W_f] < 0$ for some f and some point d agent p_i considers possible at c. Let f' be the strategy that is identical to f at all points in $\mathcal{K}_j(d)$, and hence at all points in $Tree^j_{i,d}$, but offering a payoff of 1 everywhere else. Notice that f' is a strategy (a function of p_j's local state). If p_j uses strategy f', then the best p_i can do at points not in $\mathcal{K}_j(d)$ is to break even, so $E_{Tree^j_{i,e}}[W_{f'}] \leq 0$ for $e \notin \mathcal{K}_j(d)$.

Moreover, since f and f' are identical on $Tree^j_{i,d}$, we have $E_{Tree^j_{i,d}}[W_{f'}] < 0$ by our choice of d. Since $Tree_{i,c}$ is the disjoint union of $Tree^j_{i,d}$ and sets $Tree^j_{i,e}$ with $e \notin \mathcal{K}_j(d)$, it follows that $E_{Tree_{i,c}}[W_{f'}] < 0$, and hence that $Bet(\alpha, \varphi)$ is *not* *Tree*-safe for p_i at c. □

Theorem 7: 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 $\mathcal{P}^j$-safe for p_i at c iff $\mathcal{P}^j, c \models K_i^\alpha \varphi$.

Proof: Consider the evaluation of $E_c[W_f] = E_{Tree^j_{i,c}}[W_f(\varphi, \alpha)]$ for arbitrary points c and strategies f. Since p_j has the same local state at all points of $Tree^j_{i,c}$ and f is a function of p_j's local state, p_j offers the same payoff β for a bet on φ at all points of $Tree^j_{i,c}$. Since p_i is following $Bet(\varphi, \alpha)$ at all points of $Tree^j_{i,c}$, agent p_i accepts the bet at all points of $Tree^j_{i,c}$ or rejects the bet at all such points, depending on whether $\beta \geq 1/\alpha$. If p_i rejects, then $E_c[W_f]$ is obviously 0. If p_i accepts, then p_i's profit is $\beta - 1$ at points satisfying φ and -1 at all other points, and hence $E_c[W_f] = \beta\mu^j_{i,c}(Tree^j_{i,c}(\varphi)) - 1$. (Notice that because φ is measurable with respect to $\mathcal{P}^j$, we are guaranteed that $Tree^j_{i,c}(\varphi)$ is a measurable subset of $Tree^j_{i,c}$, and hence $\mu^j_{i,c}(Tree^j_{i,c}(\varphi))$ is well-defined.)

Suppose $\mathcal{P}^j, c \models K_i^\alpha \varphi$. This means that $\mu^j_{i,d}(Tree^j_{i,d}(\varphi)) \geq \alpha$ for all points d agent p_i considers possible at c. For every point d agent p_i considers possible at c and every strategy f for p_j, therefore, we have $E_d[W_f] \geq 0$ since $\beta\mu^j_{i,d}(Tree^j_{i,d}(\varphi)) - 1 \geq (1/\alpha)\alpha - 1 = 0$ when $\beta \geq 1/\alpha$. It follows that $Bet(\varphi, \alpha)$ is safe for p_i at c.

Suppose $\mathcal{P}^j, c \not\models K_i^\alpha \varphi$. This means that $\mu^j_{i,d}(Tree^j_{i,d}(\varphi)) < \alpha$ for some point d agent p_i considers possible at c. Let f be the strategy for p_j offering a payoff of $1/\alpha$ for a bet on φ at all points p_j considers possible at d, and hence at all points of $Tree^j_{i,d}$, and 1 elsewhere. It follows that $E_d[W_f] < (1/\alpha)\alpha - 1 = 0$ for the given strategy f and the given point d agent p_i considers possible at c, and hence that $Bet(\varphi, \alpha)$ is not safe for p_i at c. □

Theorem 8: In a synchronous system, if $\mathcal{S}$ is a consistent, standard assignment, then

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

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

Proof: Theorem 7 tells us that $\mathcal{S}^j$ itself determines safe bets; and from Theorem 9(a) (proved below), it follows that if $\mathcal{S} < \mathcal{S}^j$, then $\mathcal{S}$ determines safe bets, too. This proves part (a).

To prove part (b), suppose $\mathcal{S} \not\leq \mathcal{S}^j$, which means $S_{i,c} \not\subseteq Tree^j_{i,c}$ for some agent p_i and point c. It is easy to construct a transition probability assignment τ inducing a distribution μ on the runs of $\mathcal{T}(c)$ satisfying $\mu(\mathcal{R}(S_{i,c})) > \mu(\mathcal{R}(Tree^j_{i,c}))$. To see this, notice that $S_{i,c} \not\subseteq Tree^j_{i,c}$ implies $d \in S_{i,c}$ and $d \notin Tree^j_{i,c}$ for some time k point d in $\mathcal{T}$; and if G_d is the set of points with d's global state, then $G_d \subseteq S_{i,c}$ and $G_d \cap Tree^j_{i,c} = \emptyset$ since $\mathcal{S}$ and $\mathcal{S}^j$ are state generated (they are standard). By causing τ to assign high probabilities to the edges in the path from the root of $\mathcal{T}$ to d's global state in $\mathcal{T}$, we can guarantee that $\mu(\mathcal{R}(G_d)) > 1/2$. This guarantees that $\mu(\mathcal{R}(\mathcal{S}_{i,c})) \geq \mu(\mathcal{R}(G_d)) > 1/2$, and since G_d and $Tree^j_{i,c}$ are disjoint, that $\mu(\mathcal{R}(Tree^j_{i,c})) < 1 - \mu(\mathcal{R}(G_d)) < 1/2$; so $\mu(\mathcal{R}(S_{i,c})) > \mu(\mathcal{R}(Tree^j_{i,c}))$ as desired.

Now let $\mathcal{P}$ be the probability assignment induced by $\mathcal{S}$ and τ, and let $\mathcal{P}^j$ be the probability assignment induced by $\mathcal{S}^j$ and τ. Furthermore, let G_c be the set of points with global state c, let ψ be the fact which is true precisely of the points in G_c, and let $\varphi = \neg\psi$. Since $\mathcal{L}(\Phi)$ is sufficiently rich, it follows that $\psi \in \Phi$; since $\mathcal{L}(\Phi)$ is closed under negation, it follows that $\varphi = \neg\psi \in \mathcal{L}(\Phi)$.

Since both $\mathcal{S}$ and $\mathcal{S}^j$ are standard, and hence inclusive and state generated, it follows that $G_c \subseteq S_{i,c} \cap \mathit{Tree}^j_{i,c}$. Since φ is false only at points in G_c, and since G_c is contained in both $S_{i,c}$ and $\mathit{Tree}^j_{i,c}$, it is easy to see that

$$\alpha = \mu_{i,c}(S_{i,c}(\varphi)) = \frac{\mu(\mathcal{R}(S_{i,c})) - \mu(\mathcal{R}(G_c))}{\mu(\mathcal{R}(S_{i,c}))}$$

and

$$\alpha^j = \mu^j_{i,c}(\mathit{Tree}^j_{i,c}(\varphi)) = \frac{\mu(\mathcal{R}(\mathit{Tree}^j_{i,c})) - \mu(\mathcal{R}(G_c))}{\mu(\mathcal{R}(\mathit{Tree}^j_{i,c}))}.$$

Furthermore, since $\mathcal{S}$ is uniform (it is standard), any set $S_{i,e}$ not equal to $S_{i,c}$ is disjoint from $S_{i,c}$ and hence from G_c, so $\mu_{i,e}(S_{i,e}(\varphi)) = \mu_{i,e}(S_{i,e}) = 1$ for all such sets $S_{i,e}$. It follows that $\mathcal{P}, c \models K_i^\alpha \varphi$.

On the other hand, since $\mu(\mathcal{R}(S_{i,c})) > \mu(\mathcal{R}(\mathit{Tree}^j_{i,c}))$ and $\mu(\mathcal{R}(G_c)) > 0$, it is easy to see that $\alpha > \alpha^j$. Let f be the strategy in which p_j offers a payoff of $1/\alpha$ for φ at points of $\mathcal{K}_j(c)$, and suppose p_i uses the rule $\mathit{Bet}(\varphi, \alpha)$. Clearly $W_f = W_f(\varphi, \alpha)$ is $1/\alpha - 1$ on $\mathit{Tree}^j_{i,c}(\varphi)$ and -1 off this set. Thus,

$$E_{\mathit{Tree}^j_{i,c}}(W_f) = \left(\frac{1}{\alpha} - 1\right)\alpha^j + (-1)(1 - \alpha^j) < \left(\frac{1}{\alpha} - 1\right)\alpha - (1 - \alpha) = 0,$$

which means $\mathit{Bet}(\varphi, \alpha)$ is not safe for p_i at c. $\square$

Note that the universal quantification over transition probability assignments is crucial in this proof. Given a fact φ false only at points in the intersection of $S_{i,c}$ and $\mathit{Tree}^j_{i,c}$, the proof shows that a necessary condition for $\mathcal{P}, c \models K_i^\alpha \varphi$ to imply that $\mathit{Bet}(\varphi, \alpha)$ is safe for p_i at c is that the measure of the runs through $S_{i,c}$ is less than or equal to the measure of the runs through $\mathit{Tree}^j_{i,c}$. In fact, this is a sufficient condition as well. For any given τ it may be possible to construct a set $S_{i,c} \not\subseteq \mathit{Tree}^j_{i,c}$ satisfying this condition; but the only way to satisfy this condition for all τ is to take $S_{i,c} \subseteq \mathit{Tree}^j_{i,c}$.

Theorem 9: In a synchronous system, if $\mathcal{P}'$ and $\mathcal{P}$ are consistent, standard assignments satisfying $\mathcal{P}' < \mathcal{P}$, then

(a) for every fact φ, every agent p_i, every point c, and all α, β with $0 \le \alpha \le \beta \le 1$, 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 $0 \le \alpha \le \beta \le 1$ such that

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

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

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

Proof: First we prove part (a). Suppose $\mathcal{P}', c \models K_i^{[\alpha,\beta]}\varphi$. This means $\alpha \leq \mu'_{i,d_*}(S'_{i,d}(\varphi)) \leq {\mu'_{i,d}}^*(S'_{i,d}(\varphi)) \leq \beta$ for all points $d \in \mathcal{K}_i(c)$. Choose some point $d \in \mathcal{K}_i(c)$. Since $\mathcal{P}'$ and $\mathcal{P}$ are consistent (and uniform) and satisfy $\mathcal{P}' \leq \mathcal{P}$, the set $S_{i,d}$ is the disjoint union of a collection of probability spaces $S'_{i,d_1}, \ldots, S'_{i,d_\ell}$ with $d_j \in S_{i,d} \subseteq \mathcal{K}_i(c)$, each a measurable subset of $S_{i,d}$. It follows that $S_{i,d}(\varphi)$ is the disjoint union of $S'_{i,d_1}(\varphi), \ldots, S'_{i,d_\ell}(\varphi)$. An easy computation shows that $\sum_j \mu_{i,d_*}(S'_{i,d_j}(\varphi)) \leq \mu_{i,d_*}(S_{i,d}(\varphi))$. Since $\mathcal{P}' \leq \mathcal{P}$, Proposition 5 shows that μ'_{i,d_j} can be obtained from $\mu_{i,d}$ by conditioning on S'_{i,d_j}. It follows that

$$\begin{aligned}
\mu_{i,d_*}(S'_{i,d_j}(\varphi)) &= \sup\left\{\mu_{i,d}(T') \,:\, T' \subseteq S'_{i,d_j}(\varphi),\, T' \in \mathcal{X}'_{i,d_j}\right\} \\
&= \sup\left\{\mu'_{i,d_j}(T')\mu_{i,d}(S'_{i,d_j}) \,:\, T' \subseteq S'_{i,d_j}(\varphi),\, T' \in \mathcal{X}'_{i,d_j}\right\} \\
&= \sup\left\{\mu'_{i,d_j}(T') \,:\, T' \subseteq S'_{i,d_j}(\varphi),\, T' \in \mathcal{X}'_{i,d_j}\right\} \mu_{i,d}(S'_{i,d_j}) \\
&= \mu'_{i,d_{j*}}(S'_{i,d_j}(\varphi))\, \mu_{i,d}(S'_{i,d_j}).
\end{aligned}$$

Combining the preceding statements, we have

$$\begin{aligned}
\alpha &= \sum_j \alpha\, \mu_{i,d}(S'_{i,d_j}) \\
&\leq \sum_j \mu'_{i,d_{j*}}(S'_{i,d_j}(\varphi))\, \mu_{i,d}(S'_{i,d_j}) \\
&= \sum_j \mu_{i,d_*}(S'_{i,d_j}(\varphi)) \\
&\leq \mu_{i,d_*}(S_{i,d}(\varphi))
\end{aligned}$$

A similar argument shows ${\mu_{i,d}}^*(S_{i,d}(\varphi)) \leq \beta$. Since these arguments hold for all $d \in \mathcal{K}_i(c)$, it follows that $\mathcal{P}, c \models K_i^{[\alpha,\beta]}\varphi$.

We now prove part (b). Since $\mathcal{P}' < \mathcal{P}$, it follows that $S_{i,c}$ contains two disjoint sets $S'_{i,c}$ and $S'_{i,d}$ for some agent p_i and points c and d. Let ψ be the fact true at precisely the points in the set G_c of points with c's global state, and let $\varphi = \neg\psi$. Notice that since $\mathcal{P}'$ is standard and hence state generated, G_c is contained in $S'_{i,c}$ and disjoint from $S'_{i,d}$. If $\mathcal{L}(\Phi)$ is sufficiently rich, then $\psi \in \Phi$, and hence $\varphi = \neg\psi \in \mathcal{L}(\Phi)$.

Since $G_c \subseteq S'_{i,c} \subset S_{i,c}$, the fact φ holds with probability 1 with respect to all probability spaces determined by $\mathcal{P}'$ and $\mathcal{P}$ except $S'_{i,c}$ and $S_{i,c}$. Since $\mathcal{P}' \leq \mathcal{P}$, Proposition 5 tells us that $\mu'_{i,c}$ can be obtained from $\mu_{i,c}$ by conditioning on $S'_{i,c}$. It is easy to see, therefore, that φ holds with probability

$$\alpha' = \mu'_{i,c}(S'_{i,c}(\varphi)) = \frac{\mu_{i,c}(S'_{i,c}) - \mu_{i,c}(G_c)}{\mu_{i,c}(S'_{i,c})}$$

with respect to $S'_{i,c}$, and probability

$$\alpha = \mu_{i,c}(S_{i,c}(\varphi)) = \frac{\mu_{i,c}(S_{i,c}) - \mu_{i,c}(G_c)}{\mu_{i,c}(S_{i,c})}$$

with respect to $S_{i,c}$. Since $\mu_{i,c}(S'_{i,c}) < \mu_{i,c}(S_{i,c}) = 1$, however, it is easy to see that $\alpha' < \alpha$. It follows that $\mathcal{P}, c \models K_i^{[\alpha,1]}\varphi$ but $\mathcal{P}', c \not\models K_i^{[\alpha,1]}\varphi$.

On the other hand, $\neg\varphi$ holds with probability 0 with respect to all probability spaces determined by $\mathcal{P}'$ and $\mathcal{P}$ except $S'_{i,c}$ and $S_{i,c}$. The fact $\neg\varphi$ holds with probability $1-\alpha'$ with respect to $S'_{i,c}$ and probability $1-\alpha$ with respect to $S_{i,c}$. Since $\alpha' < \alpha$, we have $1-\alpha < 1-\alpha'$; setting $\beta = 1-\alpha$, it follows that $\mathcal{P}, c \models K_i^{[0,\beta]}\neg\varphi$ but $\mathcal{P}', c \not\models K_i^{[0,\beta]}\neg\varphi$. □

Proposition 10: $\mathcal{P}^{post}, c \models K_i^{[\alpha,\beta]}\varphi$ iff $\mathcal{P}^{pts}, c \models K_i^{[\alpha,\beta]}\varphi$, for every fact φ, agent p_i, and point c.

Proof: Recall that we have fixed the class $\mathcal{A}$ of adversaries of the first type, and that for the adversary of the second type, each agent bets against itself. Consider the adversary $C \in pts(\mathcal{A})$ of the third type mapping an agent p_i and a point d to the set $S^C_{i,d}$ of points defined as follows: for every run r passing through $Tree_{i,d}$, choose a point $(r,k) \in Tree_{i,d}$ satisfying $\neg\varphi$ if such a point exists, and choose an arbitrary point $(r,k) \in Tree_{i,d}$ if all such points of r in $Tree_{i,d}$ satisfy φ. It is easy to see that the same set of runs pass through $S^C_{i,d}$ and $Tree_{i,d}$, and that a run r passes through $S^C_{i,d}(\varphi)$ iff φ is true at all points of r contained in $Tree_{i,d}$. It follows that $S^C_{i,d}(\varphi)$ and $Tree_{i,d}(\varphi)$ have the same inner measure with respect to $S^C_{i,d}$ and $Tree_{i,d}$, respectively. On the other hand, consider an arbitrary adversary $D \in pts(\mathcal{A})$ mapping p_i and d to the set $S^D_{i,d}$ (contained in $Tree_{i,d}$). Suppose the run r passes through $S^C_{i,d}(\varphi)$. It follows from the definition of $S^C_{i,d}(\varphi)$ that φ must hold at every point of r in $Tree_{i,d}$. Since $S^D_{i,d}$ must contain precisely one such point, r must pass through $S^D_{i,d}(\varphi)$ as well. It follows that the inner measure of $S^D_{i,d}(\varphi)$ must be at least the inner measure of $S^C_{i,d}(\varphi)$; thus, the infimum of $(\mu^{pts}_{i,d})_*(S^D_{i,d}(\varphi))$ — taken over all adversaries $D \in pts(\mathcal{A})$ — is precisely $(\mu^{pts}_{i,d})_*(S^C_{i,d}(\varphi)$, and we have already observed that $(\mu^{pts}_{i,d})_*(S^C_{i,d}(\varphi)$ and $(\mu^{post}_{i,d})_*(Tree_{i,d}(\varphi))$ are equal. A similar construction shows that the supremum of $(\mu^{pts}_{i,d})^*(S^D_{i,d}(\varphi))$ — taken over all adversaries $D \in pts(\mathcal{A})$ — is precisely $(\mu^{post}_{i,d})^*(Tree_{i,d}(\varphi))$. Since these statements are true for all points d agent p_i considers possible at c, we have $\mathcal{P}^{pts}, c \models K_i^{[\alpha,\beta]}\varphi$ iff $\mathcal{P}^{post}, c \models K_i^{[\alpha,\beta]}\varphi$. □

B Discussion

In this appendix, we discuss a few issues related to observations made in this paper.

B.1 The need for protocols

Although from a computer scientist's point of view, it seems quite natural to assume, as we do, that all agents in a system follow some kind of a protocol, protocols are not quite so standard in the probability theory literature. Interestingly, Shafer observes [Sha85] that it is necessary for us to think in terms of protocols if we are to make sense of "conditioning on everything an agent knows" as is done by $\mathcal{P}^{post}$. His argument, which we reproduce here, is based on *Freund's puzzle of the two aces* (see [Fre65]; other references are given in [Sha85]).

Consider a deck with four cards, the ace and deuce of hearts and spades. After a fair shuffle of the deck, two cards are dealt to p_1. Now what is the probability, according to p_2, that p_1 holds both aces? First, notice that if A, B, C, and D denote the events that p_1 holds two aces, at least one ace, the ace of spades, and the ace of hearts, respectively, then

$$Pr(A) = Pr(A \cap B) = Pr(A \cap C) = \frac{1}{6}, \quad Pr(B) = \frac{5}{6}, \quad Pr(C) = Pr(D) = \frac{1}{2}.$$

Suppose p_1 first says it holds an ace. Conditioning on this information, p_2 computes the probability p_1 holds both aces to be

$$Pr(A|B) = \frac{1/6}{5/6} = \frac{1}{5}.$$

As a result of learning p_1 holds at least one ace, the probability according to p_2 that p_1 holds both aces increases.

Suppose p_1 then says it holds the ace of spades. Conditioning on this additional information, p_2 computes the probability p_1 holds both aces to be

$$Pr(A|C) = \frac{1/6}{1/2} = \frac{1}{3}.$$

As a result of learning not only that p_1 holds at least one ace, but that it actually holds the ace of spades, the probability according to p_2 that p_1 holds both aces increases even more. Similarly, $Pr(A|D) = 1/3$.

But is this second computation reasonable? When p_2 learns B, then p_2 knows that p_1 has either the ace of spades or the ace of hearts. When p_2 learns C, then p_2 knows that p_1 definitely has the ace of spades. Is it reasonable for the probability p_2 places on event A, that p_1 holds two aces, to increase from $1/5$ to $1/3$ simply as a result of learning which of the two aces p_1 has? It seems just as reasonable to argue that the information about which ace p_1 actually has is useless, and p_2's probability of A shouldn't change upon hearing that C (or D) holds.

As Shafer points out, the right way for p_2 to update its probability of A depends on what protocol the agents are following. If the agents had agreed p_1 would first reveal whether it held an ace, and then whether it held the ace of spades, then the increase seems reasonable: if p_1 says it holds an ace, then p_2's learning p_1 does *not* hold the ace of spades causes p_2's probability that p_1 holds both aces to go down to 0; so learning that p_1 *does* hold the ace of spades should make p_2's probability go up. On the other hand, if the agents were following a protocol whereby p_1 first reveals whether it has an ace, and then, if it does, reveals the suit of one of the aces it holds, choosing between hearts and spades at random if it has both aces, then p_2's probability should not change as a result of hearing that p_1 holds the ace of spades.[20] We leave it to the reader to construct the computation trees corresponding to the two protocols described above, and to check that using $\mathcal{P}^{post}$, we do indeed get the right probabilities in each case. Again, the key point here is that we need the protocol to be completely specified in order to appropriately compute the conditional probabilities.

B.2 Safe bets and nonmeasurable facts

Recall that the statement of Theorem 7 says that for *measurable* facts, $\mathcal{P}^j$ determines safe bets against p_j. The condition of measurability is required in order for the use of expectation in the definition of a safe bet to make sense. Remember that $Bet(\varphi, \alpha)$ is safe for p_i at c if $E_d(W_f) =$

[20] Although Shafer does not mention this point, the need to assume that p_1 chooses between hearts and spades at random if it holds both aces is crucial here. For example, suppose p_2 always tells p_1 it holds the ace of hearts when it holds both aces. In this case, p_2's probability p_1 holds both aces should decrease to 0 when p_1 says it holds the ace of spades.

$E_{Tree^j_{i,d}}(W_f(\varphi,\alpha))$ is nonnegative for all points d agent p_i considers possible at c, and for all strategies f for p_j. We computed in the proof of Theorem 7 that $E_d(W_f) = \beta\mu_{i,d}(S_{i,d}(\varphi)) - 1$, where β is the payoff offered by p_j in $S_{i,d}$ ($S_{i,d}$ was actually $Tree^j_{i,d}$). In order for $\mu_{i,d}(S_{i,d}(\varphi))$ to be well defined, however, $S_{i,d}(\varphi)$ must be a measurable subset of $S_{i,d}$, which means φ must be measurable.

In fact, Theorem 7 holds for nonmeasurable facts as well, but we must first give a meaningful definition of expectation for nonmeasurable events. The intuition behind the inner and outer measures μ_* and μ^* of a measure space $(S, \mathcal{X}, \mu)$ is that $\mu_*(S')$ and $\mu^*(S')$ give upper and lower bounds on the probability of S'; if S' is actually a measurable set, of course, these bounds are equal to the actual probability. This is made precise by a classical result [Hal50] which says that if $(S, \mathcal{X}', \nu)$ extends $(S, \mathcal{X}, \mu)$ (in that $\mathcal{X}' \supseteq \mathcal{X}$ and μ and ν agree on $\mathcal{X}$), then for all sets $X \in \mathcal{X}'$, we have $\mu_*(X) \leq \nu(X) \leq \mu^*(X)$. Moreover, the bounds described by the inner and outer measure are actually attainable, in that for all subsets $X \subseteq S$, there is a probability space $(S, \mathcal{X}', \nu)$ extending $(S, \mathcal{X}, \mu)$ such that $X \in \mathcal{X}'$ and $\nu(X) = \mu_*(X)$; a similar result holds in the case of outer measure.

We want to extend these ideas to expected value. More precisely, we would like to define a notions of inner expected value and outer expected value for a "nonmeasurable" random variable X which give, respectively, lower and upper bounds on what should be the expected value of X if we were to extend the measure space as above to make X measurable. This requires some work in general, but in the special case where X takes on only two values, it can be done in a straightforward way. If the two values taken on by X are x and y, with $x > y$, then we define the inner and outer expectations of a random variable X by

$$E_*(X) = x\mu_*(X = x) + y\mu^*(X = y) \text{ and}$$
$$E^*(X) = x\mu^*(X = x) + y\mu_*(X = y).$$

It is not hard to show that these definitions agree with the expected value if the sets $X = x$ and $X = y$ are measurable, and that these values are attainable if we extend the probability space in the right way to make these sets measurable.

Notice that the random variable W_f in which we are interested in fact takes on only two values (depending on whether φ is true or false). Thus, applying these definitions, we get:

$$\begin{aligned} E_*(W_f) &= (\beta - 1)\mu_*(S_{i,d}(\varphi)) + (-1)\mu^*(S_{i,d}(\neg\varphi)) \\ &= (\beta - 1)\mu_*(S_{i,d}(\varphi)) - (1 - \mu_*(S_{i,d}(\varphi))) \\ &= \beta\mu_*(S_{i,d}(\varphi)) - 1, \end{aligned}$$

which looks very similar to the formula computed for measurable facts. Following the last two paragraphs of the proof of Theorem 7 using this formula, it is easy to see the rest of the proof holds, and hence that Theorem 7 is true using inner expectation in place of expectation in the definition of a safe bet.

Acknowledgments

Thanks to Moshe Vardi, Hagit Attiya, and Adam Grove, Daphne Koller, and a number of attendees of the IMSSS Summer Workshop on Bounded Rationality, including Robert Aumann, Elchanan Ben Porath, and Dov Samet, for stimulating discussions. We also thank our two anonymous referees for their detailed reading and comments. 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. The first author's attendance at the IMSSS Workshop on Bounded Rationality was supported by NSF Grant IRI-8814953 to the IMSSS, Stanford University.

References

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

[FH91] R. Fagin and J. Y. Halpern. Uncertainty, belief, and probability. *Computational Intelligence*, 7(3):160–173, 1991.

[FH94] R. Fagin and J. Y. Halpern. Reasoning about knowledge and probability. *Journal of the ACM*, 41(2):340–367, 1994.

[FHM90] R. Fagin, J. Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Information and Computation*, 87(1/2):78–128, 1990.

[Fre65] J. E. Freund. Puzzle or paradox? *American Statistician*, 19(4):29–44, 1965.

[FZ87] M. J. Fischer and L. D. Zuck. Relative knowledge and belief (extended abstract). Technical Report YALEU/DCS/TR-589, Yale University, 1987.

[FZ88a] M. J. Fischer and L. D. Zuck. Reasoning about uncertainty in fault-tolerant distributed systems. Technical Report YALEU/DCS/TR–643, Yale University, 1988.

[FZ88b] M. J. Fischer and L. D. Zuck. Uncertain knowledge in distributed systems. Technical Report YALEU/DCS/TR-604, Yale University, 1988.

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

[Gra78] J. Gray. Notes on database operating systems. In R. Bayer, R. M. Graham, and G. Seegmuller, editors, *Operating Systems: An Advanced Course*, Lecture Notes in Computer Science, Vol. 66. Springer-Verlag, Berlin/New York, 1978. Also appears as IBM Research Report RJ 2188, 1978.

[Hal50] P. Halmos. *Measure Theory*. Van Nostrand, 1950.

[HF89] J. Y. Halpern and R. Fagin. Modelling knowledge and action in distributed systems. *Distributed Computing*, 3(4):159–179, 1989. A preliminary version appeared in *Proc. 4th ACM Symposium on Principles of Distributed Computing*, 1985, with the title "A formal model of knowledge, action, and communication in distributed systems: preliminary report".

[HM90] J. Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *Journal of the ACM*, 37(3):549–587, 1990. A preliminary version appeared in *Proc. 3rd ACM Symposium on Principles of Distributed Computing*, 1984.

[HM92] J. Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54:319–379, 1992.

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

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

[Lew81] D. Lewis. A subjectivist's guide to objective chance. In W. L. Harper, R. Stalnaker, and G. Pearce, editors, *Ifs*, pages 267–297. Reidel, Dordrecht, Netherlands, 1981.

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

[Pag82] H. R. Pagels. *The Cosmic Code: Quantum Mechanics as the Language of Nature*. Simon and Schuster, 1982.

[PT92] P. Panangaden and S. Taylor. Concurrent common knowledge: defining agreement for asynchronous systems. *Distributed Computing*, 6(2):73–93, 1992.

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

[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):66–75, 1982.

[Sha85] G. Shafer. Conditional probability. *International Statistical Review*, 53(3):261–277, 1985.

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

[van80] B. C. van Fraassen. A temporal framework for conditionals and chance. In W. L. Harper, R. Stalnaker, and G. Pearce, editors, *Ifs*, pages 323–340. Reidel, Dordrecht, Netherlands, 1980.

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