On Ambiguities in the Interpretation of Game Trees

Joseph Y. Halpern
IBM Research Division
Almaden Research Center, Dept. K53-B2
650 Harry Road
San Jose, CA 95120–6099
halpern@almaden.ibm.com

Abstract

Piccione and Rubinstein have pointed out ambiguities in the interpretation of games of imperfect recall. They focus on the notion of time consistency, and argue that a player in a game of imperfect recall may be time inconsistent, changing his strategy despite no new information and no change in his preferences. In this paper it is argued that the apparent time inconsistency arises, in part, from an inappropriate definition of the notion. That is only part of the problem though. It is shown that in some cases the apparent time inconsistency, and, more generally, ambiguity in interpreting games of imperfect recall, stems from the fact that information sets in such games do not in general capture all the relevant features of an agent's knowledge. A model is proposed, based on earlier work in the computer science literature, that does capture all these features.

1 Introduction

In a fascinating paper, Piccione and Rubinstein [1994] (PR from now on) argue that "the model of extensive games with imperfect recall suffers from major ambiguities". They go on to say that "the difficulty interpreting the model stems from questions concerning the knowledge described by the information partition and the restrictions that the information structure imposes on the set of strategies".

At a high level, I agree with these claims completely. There are indeed problems concerning how to interpret the knowledge described by information partitions in game trees. These problems, in turn, lead to problems with the standard notion of strategy in games of imperfect recall. The underlying problem, I believe, stems from the fact that game trees are not a particularly good underlying framework to capture the knowledge of agents playing a game, particularly a game of imperfect recall. As I hope to show in this paper, by properly representing the knowledge of the agents in the game, the problems and ambiguities all disappear.

PR focus on time consistency: that is, the question of whether a player will want to change his strategy in the midst of playing a game. To illustrate this problem, PR consider what they call the "absentminded driver paradox", which they describe as follows:

Example 1.1: An individual is sitting late at night in a bar planning his midnight trip home. In order to get home he has to take the highway and get off at the second exit. Turning at the first exit leads into a disastrous area (payoff 0). Turning at the second exit yields the highest reward (payoff 4). If he continues beyond the second exit he will reach the end of the highway and find a hotel where he can spend the night (payoff 1). The driver is absentminded and is aware of this fact. When reaching an intersection he cannot tell whether it is the first or the second intersection and he cannot remember how many he has passed. ∎

The situation is described by the game tree in Figure 1. Clearly the only decision the driver

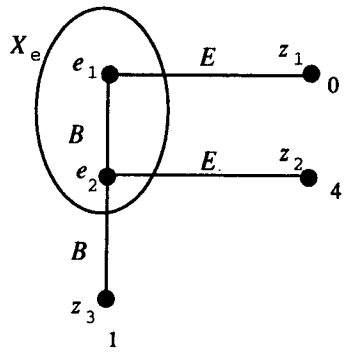

Figure 1: The absentminded driver game.

has to make is whether to get off when he reaches an exit. He cannot plan on doing different things at each exit, since, by assumption, he does not know which exit he is at when he reaches an exit. Suppose we start by considering deterministic strategies. In this case, while sitting at the bar, there is one obviously best option: not to exit. For in this case, the driver will certainly reach the end of the highway, which has payoff 1. On the other hand, if the driver decides to get off when he reaches an exit, then he will do so at the first exit, which has payoff 0.

This is the driver's reasoning at time 0, while still in the bar. Now consider what happens when he reaches the first exit. Suppose that the driver remembers the strategy he chose at the bar. Since this strategy prescribes not exiting, he should, according to PR, ascribe subjective probability $1/2$ to being at the first exit. Subjectively, he is equally likely to be at either exit. He thus concludes that it is optimal to get off, since the expected payoff of doing so is 2. This gives us time inconsistency: despite no new information and no change in his preferences, the driver is tempted to change his initial plan once he reaches an exit. PR go on to show that a similar time inconsistency arises even if the driver uses a randomized (behavioral) strategy. (Their argument is reviewed in Section 3.) This certainly seems paradoxical!

In this paper, I argue that optimal strategies are time consistent, once we remove ambiguities and use the appropriate notion of strategy. In particular, the driver should not change his mind.

Perhaps even more important than this conclusion is an understanding of what causes the appearance of time inconsistency, namely, that information sets in game trees do not constitute a good representation of what information an agent has when he needs to make a decision. Depending on what that information is, different strategies are appropriate (although in no case can the optimal strategies be viewed as time inconsistent).

To understand the issues involved, first notice that time consistency makes sense only if we assume that the agent knows his strategy. This is a point that is observed explicitly by PR (in Section 8 of their paper). Indeed, the assignment of subjective probability $1/2$ to the driver's being at the first exit depends on his knowing that he is following the strategy of never exiting. But in what sense does the agent know his strategy, when we allow for the possibility that the agent is contemplating changing that strategy? And does it make sense to assign a certain probability (such as $1/2$ in the case of the absentminded driver example) to being at a given node when this probability depends on a strategy that he may well change?

In general, an agent's decision as to whether to switch from a strategy b to a strategy b' at a given information set X depends in part on the probability the agent places on being at various nodes in X. It is not hard to show that in a game of perfect recall, this probability is independent of the strategy being followed, and depends only on X and the chance moves that were taken. On the other hand, in games of imperfect recall, the probability may depend critically on the strategy being followed (or strategies that were followed, if the agent may have changed his mind at various times in the past). Moreover, an agent who is rational and knows that he can change his mind will have extra information that may affect how he makes his decision. For example, suppose we assume that the driver remembers the most recent strategy he has used (or the sequence of strategies used). Again consider the case where the driver follows a deterministic strategy of not exiting. Now, following PR, he decides to change his mind when he reaches an exit. But before doing so, he reflects a little further and realizes that he can't possibly be at the second exit. For if he were, that means that he must have earlier been at the first exit. But the reasoning that caused him to change his mind is reasoning he could have carried out perfectly well at the first exit. Moreover, he would have reached the same conclusion at the first exit. Since he clearly hasn't changed his mind yet (this is where the assumption that the agent remembers what strategy he last follows comes in), he can't possibly have been at the first exit earlier. Realizing this, the driver changes his probability assessment to give probability 1 to being at the first exit. But then he no longer has any motivation to change his mind: he continues driving down the highway rather than exiting.

It may now seem that there is some circularity here: Once the driver realizes that he shouldn't change his mind, he no longer can assign probability 1 to being at the first exit. In that case, perhaps he should change his mind. But in that case ...

There is in fact no circularity. As we shall see, if the driver changes his mind at all, he must do so at the first exit. I present a definition of time consistency that takes this extra information into account. Essentially, this definition deals with the *absentmindedness* problem—namely, that there may be two nodes on the same path in one information set, as is the case in the absentminded driver example. According to this definition, the optimal strategy in the absentminded driver example is indeed time consistent. However, even this definition

does not get at the fundamental problem. Consider the game described in Figure 2 (this is PR's Example 2). It is not hard to show that the optimal strategy here (in the sense of giving the

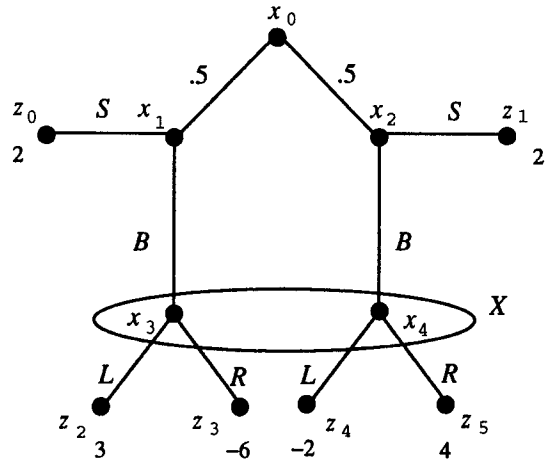

Figure 2: Another game with time inconsistency

maximum expected utility) is to choose action S at node x_1, action B at node x_2, and action R at the information set X consisting of x_3 and x_4. Call this strategy f. Let f' be the strategy of choosing action B at x_1, action S at x_2, and L at X. PR argue that if node x_1 is reached, the agent should reconsider, and decide to switch from f to f'. Under the assumption that the agent remembers the last strategy chosen (or the sequence of strategies chosen), this is correct; the agent is indeed better off (under any reasonable notion of "better off") if he switches.

The reason for the time inconsistency here is that an agent's strategy must dictate the same action at nodes x_3 and x_4, since they are in the same information set. Intuitively, since the agent cannot distinguish the nodes, he must do the same thing at both. Of course, if the agent had perfect recall, he could distinguish the nodes. In this case, the optimal strategy would essentially look like the result of switching from f to f' at x_1 without perfect recall: the agent plays L at x_3 (as he would with f') and R at x_4 (as he would with f). I claim that switching strategies ends up simulating the optimal strategy here because, by having the ability to switch strategies, the agent is able to simulate perfect recall. Among other things, PR explicitly assume "that the decision maker is not allowed to employ an external device, which is not modeled explicitly in the decision problem, to assist him in keeping the information which he would otherwise lose". However, allowing an agent to know his strategy may act as just such an external device.

This is perhaps best seen in the case of the absentminded driver. Suppose the driver is following the deterministic strategy of always exiting. If he knows his strategy, then he knows perfectly well when he is at an exit that he must be at node e_1. Thus, the driver's knowing his strategy is inconsistent with his not being able to distinguish e_1 and e_2. The same phenomenon occurs in a slightly more subtle way in the game of Figure 2. If the agent can remember the last strategy he chose to use and knows that he is rational, then when he reaches the information set $\{x_3, x_4\}$, he will know which node he is at, depending on whether the last strategy chosen

was f or f'. Again, the last strategy chosen becomes an external device that allows the agent to recover information supposedly lost.

As these examples show, an information set in a game tree may not adequately represent the information that the agent actually has. As a consequence, the restriction that a strategy must behave the same way at all nodes in an information set may be inappropriate. Of course, a strategy must be such that an agent does the same thing in all situations that he cannot distinguish. The point is that "situations that he cannot distinguish" and "nodes in the same information set" may be two quite different notions. Roughly speaking, they coincide if the agent knows his strategy, never changes it, and has perfect recall. Otherwise, they may differ.

So how can we capture an agent's information? We could appropriately modify the game tree to model the information that the agent has. I propose another solution, based on an approach introduced by Halpern and Fagin [1989]—and discussed in detail in [Fagin, Halpern, Moses, and Vardi 1995]—and extended by Halpern and Tuttle [1993] to deal with randomized actions. The idea is to distinguish the "external world" from an agent's "internal world". The game tree is a useful representation of the external world. Nodes in a game tree can be viewed as describing possible states of the external world. Information sets then represent an upper bound on what the agent can know about the external world, even assuming that she has perfect recall. To represent an agent's internal world, I assume that the agent, like the external world, is always in some *(local) state*. Intuitively, this state describes all the relevant information the agent has—about the external world, about other agents (if there are other agents in the game), about her strategy, and so on. For example, as noted in [Fagin, Halpern, Moses, and Vardi 1995], to capture the fact that an agent knows her strategy, we can encode the strategy in the agent's local state. Similarly, to capture the fact that an agent remembers what actions she has performed thus far, these actions must be encoded in the agent's state. By making the agent's state explicit, there is no doubt about what the agent knows or does not know at any point.

The analogue to a strategy in this framework is a *protocol*, which is a function from local states to actions. Protocols are meant to capture the same intuitions as strategies: what an agent does can depend only on what he knows. But now, an agent's knowledge is captured, not by the information set, but by his local state. If the information sets characterize an agent's knowledge in a game, then protocols and strategies coincide. When they do not (as is the case in the two games discussed above), then I would argue that protocols are the right notion to consider, not strategies.

In the games of imperfect recall described in PR, it is left ambiguous as to what the agent's possible states are. In the framework I present here, we are forced to be explicit about this. If we allow the agent to switch strategies, and the agent's state includes the last strategy chosen, the agent's local state will be different at nodes x_3 and x_4 in Figure 2. Since the agent's protocol is a function of his local state, not his information set, a protocol may perform different actions at these nodes. The optimal protocol in this game is indeed to choose to change strategies at node x_1 (which results in different actions being performed at x_3 and x_4) and it is indeed time consistent. My claim is that, in general, once we think in terms of local states and protocols, then all of the difficulties pointed out by PR disappear.

The rest of this paper is organized as follows. In Section 2, some basic definitions are given (which are mainly standard definitions of game theory). In Section 3, I take a closer look at the notion of time consistency, and consider various possible definitions of it and how they differ. In Section 4, I consider an alternative model of information in games, using local states and protocols, and show how it handles the problems raised by PR. I conclude with some discussion in Section 5. Proofs of all the results presented here, as well as further discussion (in particular, a careful comparison of the approach discussed in Section 3 and the "multi-self" approaches discussed by PR and by Battigalli [1995]), can be found in the full paper [Halpern 1995].

2 Basic Definitions

A game Γ is described by a game tree consisting of a finite collection of nodes partially ordered by $\prec$ (where $\prec$ is a transitive and anti-symmetric relation). As usual, we write $x \preceq y$ if $x \prec y$ or $x = y$. Intuitively, $x \preceq y$ if there is a path from x to y in the tree. Game trees are assumed to be *rooted*; that is, there is a unique node x_0 such that for all nodes y in Γ, we have $x_0 \preceq y$. Finally, we assume that $\prec$ does define a tree, in that if $x \prec y$ and $x' \prec y$, then we must have either $x \preceq x'$ or $x' \preceq x$.

For a k-player game, the nodes in a game tree can be partitioned into $k + 2$ sets denoted $C, D_1, \ldots, D_k$, and Z. Since in this paper I want to focus on single-agent games, I henceforth assume that $k = 1$, and I refer to D rather than D_1. The set C consists of *chance nodes*, where nature moves. The edges coming out of nodes in C are labeled by the probability of nature taking that move. For example, in Figure 2, x_0 is a chance node; nature moves from x_0 to x_1 with probability .5. The set D consists of *decision nodes* where the player must move. D is further partitioned into *information sets*. (The information sets are described by ellipses in the game tree.) For example, in Figure 1, e_1 and e_2 are in the same information set. Intuitively, if two nodes are in the same information set, then the player cannot distinguish them. Exactly what "cannot distinguish" means is one of the major issues dealt with in this paper. At each node x in D, the player can choose among some set of actions, denoted $A(x)$. The edges coming out of x are labeled by these actions. We assume that the same set of actions can be performed at each node in a given information set. That is, if x and y are both in information set X, then $A(x) = A(y)$. For example, since e_1 and e_2 are both in information set X_e in Figure 1, we have $A(e_1) = A(e_2) = \{B, E\}$. Thus, we can write $A(X)$ to denote the actions that can be taken at an information set X. The set Z consists of the set of terminal nodes in Γ; that is, those nodes z for which there does not exist z' such that $z \prec z'$. With each node $z \in Z$ is associated some a utility $u(z)$; $u(z)$ can be thought of as the payoff for reaching node z. For example, in Figure 1, $u(z_1) = 0$ and $u(z_2) = 4$.

Roughly speaking, an agent in game Γ has *perfect recall* if he always remembers what actions he has taken and what he knew previously. Formally, this is captured by associating with each node x in the game tree the sequence $exp(x)$ (for the *experience* of the agent at node x) of actions taken by the agent and information sets gone through by the agent in going from the root to x. For example, in Figure 1, $exp(e_2) = \langle X_e, B, X_e \rangle$, while in Figure 2,

$exp(x_3) = \langle\{x_1\}, B, \{x_3, x_4\}\rangle$. I omit the formal definition here. A game of perfect recall is one where, for every information set X, we have $exp(x) = exp(x')$ for all nodes $x, x' \in X$. Thus, the game in Figure 1 is not a game of perfect recall, since $exp(e_1) \neq exp(e_2)$. The game in Figure 2 is not a game of perfect recall either, since $exp(x_3) \neq exp(x_4)$.

We can similarly associate with each node x two other sequences, denoted $exp'(x)$ and $exp''(x)$. The sequence $exp'(x)$ consists of the sequence of information sets gone through by the agent (but not the sequence of actions), while $exp''(x)$ consists of the sequence of information sets without consecutive repetitions. Thus, for example, in Figure 1, we have $exp'(e_2) = \langle X_e, X_e\rangle$, while $exp''(e_2) = \langle X_e\rangle$, with the second X_e omitted. PR say a game has *perfect recall of information sets* if, for every information set X and $x, x' \in X$, we have $exp'(x) = exp'(x')$. Similarly, I say that Γ has *partial recall (of information sets)* if, for every information set X and $x, x' \in X$, we have $exp''(x) = exp''(x')$.[1] Note that the absentminded driver example exhibits partial recall, although not perfect recall of information sets. The game in Figure 2 does not exhibit partial recall (and, *a fortiori*, does not exhibit perfect recall of information sets either).

PR say that a game exhibits *absentmindedness* if there is an information set X with two nodes $x, x' \in X$ such that $x \prec x'$. Clearly a game that exhibits absentmindedness cannot be a game of perfect recall. The absentminded driver example of Figure 1 is a game that exhibits absentmindedness.

Intuitively, a *behavior strategy* specifies the action that an agent takes at each node $x \in D$. The choice of action may be randomized. Formally, a behavior strategy b assigns to each node $x \in D$ a probability distribution $b(x)$ over the actions in $A(x)$. For notational convenience, this distribution is often denoted b_x. (Of course, b is deterministic if b_x assigns probability 1 to some action in $A(x)$.) If nodes x and y are in the same information set X, we assume that $b_x = b_y$, since the agent is not supposed to be able to tell which of these nodes he is at. Thus, we can write b_X to denote the distribution on the actions in $A(X)$. For reasons that will shortly become clear, it is also useful to have *generalized behavior strategies*, where the requirement that $b_x = b_y$ for nodes x and y in the same information set is dropped. We can think of b_x as a random device (e.g., a coin toss) that is activated when node x is reached. For example, if at node x an agent has two possible actions, L and R, and $b_x(L) = .6$, then at node x, the agent tosses a coin which lands heads with probability .6, and chooses action L if the coin lands heads, and action R otherwise. If x and y are distinct nodes such that $b_x = b_y$, then the coin is tossed independently at x and y; the outcome at x does not affect the outcome at y. Given a (generalized) behavior strategy b and a node x in the tree, let $p_b(y|x)$ denote the probability of reaching node y starting at node x and using strategy b. Clearly, if there is no path from x to y in Γ, we must have $p_b(y|x) = 0$. If there is a path, then $p_b(y|x)$ is just the probability of choosing the (unique) sequence of actions that lead from x to y, according to behavior strategy

[1] Interestingly, the notion of partial recall of information sets is actually closer in spirit to what is called perfect recall in the computer science literature [Fagin, Halpern, Moses, and Vardi 1995; Halpern and Vardi 1986] than the standard definition of perfect recall (in terms of *exp*) used in the game theory literature. Roughly speaking, $exp''(x)$ is meant to capture the intuition that the agent is not aware of time passing. Discussion of and more motivation for the computer science definition is given in Footnote 5.

b. Finally, let $p_b(y)$ be an abbreviation for $p_b(y|r)$, where r is the root of the tree.

3 A Closer Look at Time Consistency

In this section, I want to discuss the notion of time consistency in greater detail. Roughly speaking, a behavior strategy b is said to be time consistent if, whenever an agent reaches an information set X in the game tree, then the agent does not consider some other strategy b' to be better than b, given that he has reached X. To formalize this intuition, we need to have some way of evaluating the relative "goodness" of two strategies b and b' at an information set X.

As usual, goodness is evaluated in terms of expected utility. For each node x in X, it is easy to evaluate the expected utility of a strategy b, starting at x; it is simply $EU(b;x) = \sum_{z \in Z} p_b(z|x)u(z)$. Of course, the expected utility of b, denoted $EU(b)$, is just $\sum_{z \in Z} p_b(z)u(z)$. If the agent also has some subjective probability $\mu_b(\cdot|X)$ on the relative likelihood of the nodes in X, then it seems reasonable to take the expected utility of using b starting in X to be

$$\sum_{x \in X} \mu_b(x|X)EU(b;x).$$

In fact, this definition is not as reasonable as it may at first appear; see [Grove and Halpern 1995] for a discussion of this issue. However, let us follow PR and accept it for now. (In any case, the approach taken in this section is consistent with the analysis in [Grove and Halpern 1995].) I will assume (just as PR do throughout most of their paper) that $\mu_b(\cdot|X)$ is *consistent with* b, so that if $p_b(X) > 0$, then $\mu_b(x|X) = p_b(x)/\sum_{x' \in X} p_b(x')$. Thus, the relative weight of nodes in X is the relative likelihood of reaching these nodes according to the strategy b. In the game theory literature, it is standard to consider $\mu_b(\cdot|X)$ as a family of distributions, one for each information set X in the game (see, for example, [Battigalli 1995; Myerson 1991; Piccione and Rubinstein 1994]). It is more convenient here to think of this family as being generated by conditioning on one distribution μ_b on the nodes in the tree, where $\mu_b(x) = p_b(x)/\sum_{x'} p_b(x)$. As pointed out in PR, we can think of this distribution as being determined by the following experiment: "Play the game over and over, using strategy b. As soon as a terminal node is reached, start over again. In what fraction of the first N units if time (think of N as large; formally, it is taken to infinity) is the player at node x?" It is not hard to show that this likelihood is characterized by μ_b. The advantage of considering μ_b, rather than a family $\mu_b(\cdot|X)$, is that it allows us to condition on sets other than information sets. This will play an important role later in the paper.

With these definitions in hand, we can now state the PR definition of time consistency:

Definition 3.1: A behavior strategy b is *PR time consistent* if for all information sets X such that $p_b(X) > 0$ and all strategies b', we have

$$\sum_{x \in X} \mu_b(x|X)EU(b;x) \geq \sum_{x \in X} \mu_b(x|X)EU(b';x).$$

One way of understanding this definition (which is not necessarily the way PR intend it to be understood) is the following: assume that the agent is using strategy b, and at a given node x in information set X the agent contemplates whether he would like to deviate from the strategy. This is to be viewed as idle contemplation, in the sense that even if he decides he would prefer to switch, he does not actually do so; he continues to use b no matter what the result of the contemplation. Of course, when the agent is asked at node x whether he would like to deviate, all he knows is that he is in information set X and that he is using strategy b. Thus, $\mu_b(x|X)$ is a reasonable estimate of being at x. If we accept $\mu_b(x|X)$ as a reasonable assessment of being at x given that information set X has been reached, then b is PR time consistent precisely if, for each information set X that is reached with positive probability, there is no strategy that has greater expected utility than b, given that X has been reached.

But is $\mu_b(x|X)$ a reasonable assessment of being at x given that X has been reached? I claim that it is not if we do not view the reassessment process as just idle contemplation. (And it seems clear from their paper that PR do not intend for it to be viewed as idle contemplation.) Suppose that if the agent decides at x that he prefers b' to b, then he actually switches strategies. Under what circumstances should the agent want to switch from b to b' at information set X? Switching at information set X amounts to using the strategy that I'll denote (b, X, b'); that is, b is used at all nodes except those in X or below X in the tree; at these nodes, b' is used. Note that even if b is a strategy, (b, X, b') may be a generalized strategy if Γ is not a game of partial recall. (An example of this phenomenon is provided in [Halpern 1995].)

An agent that is contemplating switching from b to b' at X must evaluate the relative benefits of doing so. At first glance, a variant of the definition above that uses $\mu_{(b,X,b')}$, not μ_b, captures this intuition. But there is another subtlety here. If an agent decides that b' is a better choice—indeed, the best choice—at information set X, not b, and the agent can actually change strategies, then the agent will make the change from b to b' the first chance that he gets. *Provided he is aware that he has made this change once he has made it* then the agent can conclude that he must be at the *upper frontier* of X; where the upper frontier of an information set X consists of all those nodes $x \in X$ such that there is no node $x' \in X$ that precedes x on some path from the root.[2] Let $\hat{X}$ denote the upper frontier of X. For example, in the absentminded driver game, $\hat{X}_e = \{e_1\}$. In light of the discussion above, the following definition captures a notion of consistency that seems appropriate for agents who are aware of what strategy they are following:

Definition 3.2: A behavior strategy b is *gt ("game tree") time consistent* if for all information sets X such that $p_b(X) > 0$ and all strategies b', we have

$$\sum_{x \in \hat{X}} \mu_b(x|\hat{X}) EU(b; x) \geq \sum_{x \in \hat{X}} \mu_{(b,X,b')}(x|\hat{X}) EU(b'; x).$$

[2] As Bart Lipman [private communication] has pointed out, it may seem that this argument breaks down if we allow randomization. That is, we can consider allowing the agent to decide to switch from b to b' with some probability α. In that case, the switch from b to b' may not happen at the upper frontier. On the other hand, what happens is that the agent switches from b to the strategy we can denote $(1 - \alpha)b + \alpha b'$ (continue using strategy b with probability $1 - \alpha$ and switch to b' with probability α), and this switch must happen at the upper frontier. Since $(1 - \alpha)b + \alpha b'$ is equivalent to a behavior strategy, the same analysis as given here again applies.

Thus, the difference between gt time consistency and PR time consistency is the use of $\hat{X}$ rather than X and the use of $\mu_{(b,X,b')}$ on the right-hand side rather than μ_b. I could have also used $EU((b, X, b'); x)$ instead of $EU(b'; x)$, but since (b, X, b') and b' lead to the same actions for nodes in X or below X, it should be clear that these expected utilities are equal. In fact, it follows from Proposition 3.3 that we also have $\mu_b(\cdot|\hat{X}) = \mu_{(b,X,b')}(\cdot|\hat{X})$. Thus, the really significant difference is the use of $\hat{X}$ instead of X. To repeat, the key point here is that if it is rational to switch from b to b' at information set X, and the agent is rational, the agent can conclude that the switch must be made at the upper frontier of the information set. By conditioning on $\hat{X}$ instead of X, this conclusion is also reflected in his belief function.

I would argue that gt time consistency captures the intuition of time consistency better than PR time consistency. How different are the two notions? As is shown below, they differ only in games that exhibit absentmindedness. These are precisely the games where $\hat{X}$ may differ from X. For example, in the absentminded driver game of Figure 1, let d be the deterministic strategy where the driver never exits, and d' the deterministic strategy where he always exits. It is easy to see that $\mu_d(e_1) = \mu_d(e_2) = 1/2$, while $\mu_{d'}(e_1) = 1$ and $\mu_{d'}(e_2) = 0$. According to the PR definition, the driver who starts by using strategy d should switch to d' at information set X_e, since the expected utility of switching to d' is $2\ (0 \times 1/2 + 4 \times 1/2)$ while the expected utility of sticking with d is only 1. This is a correct calculation only if the driver deciding that d' is better than d at e_1 does not result in the driver exiting at e_1. Under these circumstances, it seems reasonable for the agent to assign equal probability to being at e_1 or e_2. On the other hand, if the agent truly switches strategies whenever he reaches information set X_e, then he actually does so at the point e_1, and so his expected utility is 0, not 2, just as would be calculated using the gt notion of time consistency. Put another way, if the agent just contemplates making changes (but never makes them), then he could contemplate them equally well at every point in an information set. But if he actually makes the change, then it must get made at the upper frontier.

Proposition 3.3: *Suppose Γ a game, X is an information set, and b and b' are strategies such that $p_b(X) > 0$ and $p_{b'}(X) > 0$.*

(a) If Γ is a game of perfect recall, then $\mu_b(\cdot|X) = \mu_{b'}(\cdot|X)$.

(b) If Γ is a game with no absentmindedness, and $b(y) = b'(y)$ for all nodes $y \in D$ that precede some node in X, then $\mu_b(\cdot|X) = \mu_{b'}(\cdot|X)$.

(c) If Γ is a game with absentmindedness, and $b(y) = b'(y)$ for all nodes $y \in D$ that precede some node in $\hat{X}$, then $\mu_b(\cdot|\hat{X}) = \mu_{b'}(\cdot|\hat{X})$.

This result has important implications for the notion of time consistency. Part (a) tells us that in games of perfect recall, the agent does not need to know the strategy used to get to information set X to evaluate which strategy he should use to continue, since $\mu_b(x|X)$ is independent of b. It follows from part (b) that in games of imperfect recall that do not exhibit absentmindedness, we have $\mu_b(\cdot|X) = \mu_{(b,X,b')}(\cdot|X)$. Since in such games we also have $\hat{X} = X$ for all information sets X, we immediately get

Corollary 3.4: *In games with no absentmindedness, a strategy is PR time consistent iff it is gt time consistent.*

Notice, however, that in games of imperfect recall, even with no absentmindedness, $\mu_b(\cdot|X)$ may in general depend on b. For example, it is easy to see that in the game described in Figure 2, for the strategies f and f' discussed in the introduction, $\mu_f(x_3|X) = 0$ while $\mu_{f'}(x_3|X) = 1$. That means that for games of imperfect recall, to evaluate which strategy to use at an information set X (and to make sense of the notion of time consistency), the agent must know what strategy he has been using up to (but not including) X. Note that the agent does not have to know what he will do at or after X.

For games of imperfect recall with absentmindedness, as we shall see, PR time consistency and gt time consistency do not agree in general. However, notice that by part (c) of Proposition 3.3, we could replace b by (b, X, b') in the definition of gt time consistency with no change, since these two strategies agree on all nodes that precede some node in X. Moreover, when using gt time consistency, it suffices for the agent to know what his strategy has been up to (but not including X). This is not true in the case of PR time consistency. Even if b and b' agree on all nodes in D that precede the nodes in X, we may have $\mu_b(x|X) \neq \mu_{b'}(x|X)$ for some $x \in X$. To see this, consider the absentminded driver example. If d is the strategy of never exiting and d' is the strategy of always exiting, then $\mu_d(e_1|X_e) = 1/2$ while $\mu_{d'}(e_1|X_e) = 1$, although d and d' agree on the empty set of nodes that precede X_e. This means that PR time consistency makes sense only if the agent knows, not only what strategy he has used up to an information set X, but what strategy he is going to use at X itself. Again, this suggests that it may be inappropriate to use PR time consistency if the agent can actually change strategies, as opposed to just idly contemplating the possibility.

In the case of perfect recall, it is well known that a strategy is optimal iff it is PR time consistent. In fact, PR prove that this is true in games with perfect recall of information sets, provided that there is no absentmindedness.[3] Since, by Proposition 3.4, gt time consistency and PR time consistency coincide in games with no absentmindedness, this is true for gt time consistency as well. In fact, an even stronger result holds for gt time consistency.

Theorem 3.5: *If Γ is a game of partial recall, then b is optimal iff b is gt time consistent.*

Since the absentminded driver game is one of partial recall, it follows from Theorem 3.5 that the only gt time consistent strategy in that game is the optimal strategy of exiting with probability $1/3$. On the other hand, as PR show, the only PR time consistent strategy in that game is to exit with probability $5/9$.

Theorem 3.5 follows from a characterization of gt time consistency that may be of independent interest.

[3]Actually, they show it for a condition slightly weaker than perfect recall of information sets, but they do restrict to games with no absentmindedness.

Definition 3.6: If Y is a set of nodes in D, then a strategy b' is *identical to b off Y*, written $b \approx_Y b'$, if $b(x) = b'(x)$ for all $x \in D - Y$. A strategy b in *optimal on Y* if $EU(b) \geq EU(b')$ for all b' such that $b \approx_Y b'$.

Proposition 3.7: *A strategy b is gt time consistent in Γ iff for each information set X such that $p_b(X) > 0$, b is optimal on $R(X)$, where $R(X)$ consists of all the agent's nodes reachable from some node in X.*

Once we move beyond games of partial recall, the optimal strategy may not be gt time consistent. Consider the game in Figure 2 (which is not a game of partial recall). In this game, PR time consistency and gt time consistency coincide, and they do not agree with optimality. At the node x_1, it is clear that the agent should switch from f to f'. He can do better than his optimal strategy by switching. In fact, even in games of partial recall, if the agent can switch strategies, he may be able to do better than his optimal strategy. In particular, as we shall see in Section 4.2, the absentminded driver can improve on his optimal strategy if he is allowed to switch strategies. What is going on here?

As I observed in the introduction, an agent who (quite rationally) decides to switch from f to f' at node x_1 ends up playing the optimal strategy for the game where x_3 and x_4 are distinguishable: that is, he chooses action B at both x_1 and x_2, L at x_3, and R at x_4. This suggests that the ability to switch strategies at x_1 gives the agent the ability to pass on information that allows him to distinguish x_3 from x_4. The information sets are simply not capturing the agent's knowledge in this game. This intuition is made precise in the Section 4.1, where an arguably clearer model of the agent's information is presented. In the full paper, I also consider another approach to avoiding the problem of "passing information down" considered by PR and Battigalli [1995]; for lack of space, this approach is not discussed here.

4 Representing Games as Systems

An information set in a game tree is supposed to model what each agent knows about his choices at the point when he makes his move. However, since an information set consists of a set of nodes in the game tree, at best, it can only model what an agent knows about the moves that have been made. By putting nodes x and y in the same information set, we are trying to capture a situation where an agent does not know whether the sequence of actions that has occurred is the one that led to x or the one that led to y. What this means is that information sets cannot model information (or lack of it) about future moves, or, indeed, about anything in the future. (As Rubinstein [1991] observes, it is difficult to model considerations such as "I know that if I reach the second intersection I will have doubt of 10% that I am at the first intersection".) Nor can information sets model information about the strategy that other agents are using (including information about the strategy that the agent herself is using).[4] Indeed, traditionally, an agent's

[4]Of course, in a given information set, an agent does know that no strategy that did not lead to that information set is possible.

information sets consist only of nodes where the agent is about to move. (This is not a universal assumption; for example, it is not made by Rasmusen [1989].) Nodes where agent 1 does not move are not part of any information set for agent 1. The traditional notion of knowledge says that an agent knows a fact if it is true at all nodes in his information set. This means that we cannot make sense out of statements such as "agent 1 knows that agent 2 knows that agent 1 has moved left", let alone "it is common knowledge that agent 1 has moved left", because a node that is in some information set for agent 1 are not in any of agent 2's information sets. Battigalli and Bonnano [1995] propose one particular approach for making sense of such statements in games of perfect recall, but this is clearly not a general solution to the problem.

So how can we model such considerations? There is no difficulty doing so using the standard states-of-the-world approach first used in the economics literature by Aumann [1976] (which actually is a variant of the standard possible-worlds model for knowledge in the philosophical literature that goes back to Hintikka [1962]; see [Fagin, Halpern, Moses, and Vardi 1995] for discussion). In this approach, each state in a state space Ω is a complete description of the world, which includes what happened in the past and what will happen in the future, the agents' beliefs and knowledge, and so on. The trouble with this approach is that, while it does a good job of capturing an agent's knowledge, it does not do such a good job of describing the play of the game—who moves when, and what the possible moves are. Moreover, because time is not explicit in this approach, it becomes difficult to model statements such as "I know now that after I move my opponent will not know . . . ".

I describe in the next subsection (a slightly simplified version of) an approach that can be viewed as combining features of both game trees and the states-of-the-world approach, that goes back to [Halpern and Fagin 1989], and has been used quite successfully in dealing with computer science problems [Fagin, Halpern, Moses, and Vardi 1995]. In this approach, a game is represented as a multi-agent *system*. In the description of the system, the actual play of the game is distinguished from what goes on in the agent's mind.

4.1 The Framework

To describe the agent's state of mind, we assume that, at any point in time, the agent is in some *state*. Occasionally this is called a *local state*, to distinguish it from a *global state*, which is defined below. The local state is essentially what is called an agent's *type* in the game theory literature. Intuitively, it encapsulates all the information to which the agent has access. Deciding how to model the state can be quite a nontrivial issue. In a poker game, a player's state might consist of the cards he currently holds, the bets made by the other players, any other cards he has seen, and any information he has about the strategies of the other players. Alternatively, a forgetful player may not remember all the details of the bets made by the other players; his state would reflect this.

To describe the external world, we use an *environment*, which is also in some state at any point in time. Roughly speaking, the environment's state describes everything relevant to the system that is not part of the agents' states. For example, when describing a game, we can take

the environment's state to consist of the sequence of actions that have been taken up to a certain point. If we do this, we can essentially identify the possible environment states with the nodes in the game tree.

The system as a whole can be described by a *global state*, a tuple of the form $(\ell_e, \ell_1, \ldots, \ell_n)$, where ℓ_e is the environment's state, and ℓ_i is agent i's state, $i = 1, \ldots, n$. A global state describes the system at any given point in time. We are typically interested in dynamic systems that change over time. A *run* is a function from time (which is taken for simplicity to range over the natural numbers) to global states. Intuitively, a run is a complete description of how the system's global state evolves over time. For example, when analyzing a game, a run could be a particular play of a game. Thus, if r is a run, $r(0)$ describes the initial global state of the system, $r(1)$ describes the next global state, and so on. A *point* is a pair (r, m) consisting of a run r and time m. If $r(m) = (\ell_e, \ell_1, \ldots, \ell_n)$, let $r_i(m) = \ell_i$. Thus, $r_i(m)$ is agent i's local state at the point (r, m).

Finally, a *system* is formally defined to be a set of runs. Intuitively, a system is being identified with its set of possible behaviors. Thus, for example, the game of bridge can be identified with all the possible games of bridge that can be played (where a run describes a particular game, by describing the deal of the cards, the bidding, and the play of the hand).

Two things that are absent from this definition are actions and information sets. Information sets in fact do not have to be specified exogenously; they can be reconstructed from the local states. Given a system, that is, a set $\mathcal{R}$ of runs, we can define an equivalence relation on the points in $\mathcal{R}$. The point (r, m) is *indistinguishable from* (r', m') *by agent* i, denoted $(r, m) \sim_i (r', m')$, if $r_i(m) = r'_i(m)$. Thus, two points are indistinguishable by agent i if agent i has the same local state at each point. Clearly $\sim_i$ is an equivalence relation. The $\sim_i$ relations can be viewed as defining information sets. However, note that even a point where agent i does not move is in an information set for agent i.

Although actions are not part of the formal definition of a system, they often do appear (either implicitly or explicitly) in the global state of the system. That is because we typically think of the runs of the system as being generated by the agents and the environment performing some actions. For example, in modeling the game of bridge, we may well put actions such as bidding into the global state. If the agent knows what action was taken, it would be encoded in his local state. Otherwise, we can encode it in the environment's state if the action is considered relevant to the description of the system.[5] In [Fagin, Halpern, Moses, and Vardi 1995], there is a notion of a *context* in which the agents are embedded (i.e., the context in which the game

[5]With this background, I can explain how perfect recall is defined in the computer science literature [Fagin, Halpern, Moses, and Vardi 1995; Halpern and Vardi 1986]. Borrowing the notation from Section 2, define $exp''(r, m)$ to be the sequence of local states that the agent has gone through, with consecutive repetitions omitted. An agent is said to exhibit perfect recall if, whenever $(r', m') \sim_i (r, m)$, then $exp''(r, m) = exp''(r', m')$. Intuitively, this definition of perfect recall says that the agent remembers everything he ever knew. It does not say that he remembers everything he ever did. He may not remember his actions if his actions were not encoded in his local state to begin with. The reason that consecutive repetitions are omitted is that if an agent has the same local state at a sequence of consecutive states, this just means that time has passed without the agent being aware of it (or of anything else that may have happened, including actions that he has performed). Again, this is not counted against the agent having perfect recall of everything he knew.

is played), which includes the actions that can be taken and what their effects are. Rather than going through all the definitions here, I present a simplified version, sketching an approach for going from a game tree to a system.

To start with, we must decide how to model the states of the agent and the environment. Of course, there are many ways to do this. Here is one approach that should be useful for many games of interest: Given a game Γ, let the environment states be the nodes in Γ. (Note that a node in the game tree can be identified with the sequence of actions taken to reach that node, so this is how actions are encoded in the global state.) The agent's state is a tuple of the form $(X, \ldots)$, where X is a set of nodes in the tree (intuitively, this is the set of nodes that the agent considers possible), and the other components in the agent's local state (represented by the "...") incorporate whatever other relevant information the agent may have (for example, about his protocol, or protocols that other agents may be using). In a global state $(x, (X, \ldots))$, we require that $x \in X$. Intuitively, this means that the actual node in the tree (x) is one of the nodes that the agent considers possible. This requirement essentially ensures that the agent's beliefs about the current node in the tree are correct. Although I make this assumption here, it is certainly not a necessary assumption. By dropping it, we allow the agent to have false beliefs. For the purposes of this paper, X is further constrained so that if $x \in D$, then X is the information set containing x. Notice, however, this may not always be desirable. For example, it may be more convenient to model the fact that the agent has forgotten some information by appropriately modifying the agent's local state (that is, by by changing the set of nodes that the agent considers possible from X to some larger set X', which is not an information set) than by changing the information sets in the game tree.

Once we have settled on the form of the global state, we need to determine how to generate the runs. To do this, we associate with each global state g a set of actions $A(g)$ allowable at g, where an action is a function from global states to global states. What actions are possible at the global state $(x, (X, \ldots))$ and how do they change the global state? That depends on whether x is in D, C, or Z:

- If $x \in D$, we could take the possible actions to be just those in $A(x)$. However, as we shall see, it is sometimes useful to allow more general actions, which can affect the other components of an agent's state. Thus, we assume that actions in $A(x, \ell)$ have the form $(\ldots, a)$, where $a \in A(x)$. (Although, as we shall see, not all the actions in $A(x)$ are necessarily in $A(x, \ell)$.) After performing action $(\ldots, a)$, the global state becomes $(a(x), \ell')$, for some appropriate local state ℓ'. I further assume that $A(x, \ell)$ depends only on the agent's local state ℓ, so that $A(x, \ell) = A(x', \ell)$ if $x, x' \in D$.

- If $x \in C$, then $A(x, (X, \ldots)) = \{c_1, \ldots, c_k\}$, where k is the outdegree of the node x in the game tree. Performing action c_j in (x, ℓ) results in a global state of the form $(c_j(x), \ell')$, where $c_j(x)$ is the jth successor of x in the tree, in some fixed ordering of x's successors.

- If $x \in Z$, then $A(x, \ell) = \emptyset$; no actions are possible at terminal nodes.

Once we have settled on a set $\mathcal{G}$ of global states as above and a function A associating with each global state $g \in \mathcal{G}$ a set $A(g)$ of allowable actions in g, we can define a system as follows: Let $\mathcal{G}_0$, the *initial states of the system*, consist of all those global states in $\mathcal{G}$ of the form (x, ℓ), where x is the root of Γ. A run r is *consistent with* $(\mathcal{G}, \mathcal{G}_0, A)$ if $r(0) \in \mathcal{G}_0$ (so that the initial state is a legal initial state, and $r(m+1) = a(r(m))$ for some $a \in A(r(m))$ (so that the global state at time $m+1$ is the result of applying an allowed action to the global state at time m).[6] The system characterizing the game (under this representation of the game) consists of all runs consistent with $(\mathcal{G}, \mathcal{G}_0, A)$.

A *protocol* for an agent in this setting is simply a function from that agent's local states to a distribution over allowable actions. It is the straightforward analogue of the notion of strategy, and captures the intuition that what an agent does can depend only on his information. At two points where the agent has the same information, the agent must do the same thing. When it comes to modeling a game, a protocol may not coincide with the standard notion of strategy in a game tree. That is because, as we shall see, the agent's local states may impose a finer partition on points than the information state in a game tree.

4.2 The Absentminded Driver Reconsidered

Suppose we try to apply this framework to the absentminded driver example. What are the states and actions? That depends. First consider the case where the driver knows his strategy and never changes it. In this simple setting, a strategy can be represented by a real number α in the interval [0,1], where α represents the probability of exiting. For simplicity, I identify 1 with the deterministic strategy of always exiting and 0 with the deterministic strategy of never exiting. We can now use the approach suggested at the end of the previous subsection. Since the driver is not supposed to be able to distinguish the nodes e_1 and e_2, the set L_d of possible local states for the driver can be taken to be pairs (w, α), where w is either X_e, $\{z_1\}$, $\{z_2\}$, or $\{z_3\}$. Intuitively, in local state $X_e = \{e_1, e_2\}$, the driver does not know whether he is at node e_1 or at node e_2. Once we do this, we can proceed as above to construct the set of global states, which have the form $(x, (X, \alpha))$. The initial states are precisely those of the form $(e_1, (X_e, \alpha))$.

What about actions? In a global state where the driver's local state is (X_e, α), with $\alpha \neq 0, 1$, the allowable actions are, as expected, B and E. If $\alpha = 0$, then the only allowable action is B, since we are identifying $\alpha = 0$ with the deterministic strategy of never exiting; similarly, if $\alpha = 1$, then the only allowable action is E. The actions have the obvious effect on local states: for example, $E(e_2, (X_e, \alpha)) = (z_2, (\{z_2\}, \alpha))$ and $B(e_1, (X_e, \alpha)) = (e_2, (X_e, \alpha))$. Notice that the α component of the driver's local state is unaffected by the action. Thus, the agent never changes his strategy, and always knows it.

It is now easy to generate the set of runs consistent with this system. For $\alpha \neq 0, 1$, there are three runs where the driver uses strategy α. This collection of three runs can be viewed as a

[6] Since no action is allowed at a terminal node, all runs will be finite. This is in contrast to [Fagin, Halpern, Moses, and Vardi 1995; Halpern and Fagin 1989], where all runs are assumed to be infinite. This is a minor distinction; for example, I could have assumed that at a terminal node, the agent takes a "no-op" action, which leaves the global state unchanged.

copy of the game in Figure 1, where the runs correspond to the three paths in the tree ending in a terminal node. Since $(e_1, (X_e, \alpha)) \sim_1 (e_2, (X_e, \alpha))$, the driver cannot distinguish whether he is at node e_1 or node e_2, as we would expect. On the other hand, if α is either 0 or 1, there is only one run consistent with the (deterministic) strategy α. Moreover, if $\alpha = 1$, the driver knows when he is at the node e_1 that he cannot be at the node e_2, since the global state $(e_2, (X_e, 1))$ is not reachable from $(e_1, (X_e, 1))$. (Indeed, the global state $(e_2, (X_e, 1))$ does not appear on any run in the system.) Thus, the framework is able to capture the fact that if the driver is using the deterministic strategy of always exiting, he knows perfectly well when he is at node e_1 that he is at node e_1.

Now what happens if the driver is allowed to change strategies? For one thing, we need to decide what the driver remembers about his previous strategies. For definiteness, let us assume that the driver just remembers the last strategy he has used (but not the ones before that). In this case, the set of global states can be taken to be the same as it was before. The way we capture the fact that the agent can change strategies is not via the global states, but via the actions. Since the agent can now change strategies as well as making a move in the game, the set of actions consists of all pairs of the form (α, a), where $\alpha \in [0, 1]$ and $a \in \{B, E\}$, except for $(0, E)$ and $(1, B)$. The interpretation of the action (α, a) is that the driver adopts strategy α and takes action a. I am requiring that the action taken be consistent with the strategy adopted, which is why $(0, E)$ and $(1, B)$ are disallowed. As before, if the driver's local state is (z_j, α), $j = 1, 2, 3$, then there are no allowable actions. If the driver's state is (X_e, α), then all actions are allowable, and have the obvious effect. For example, $(1/3, E)(e_2, (X_e, \alpha)) = (z_2, (\{z_2\}, 1/3))$ and $(1/3, B)(e_1, (X_e, \alpha)) = (e_2, (X_e, 1/3))$.

Note that, under these assumptions, the driver does *not* have enough information to compute the expectations required in either PR time consistency or gt time consistency, since he does not necessarily know all the strategies he has used up to the present point (cf. Proposition 3.3). Nevertheless, he does know that if he changes strategy, he does so first at the upper frontier of an information set. More precisely, if he follows a deterministic protocol for deciding whether and when to change strategy, if a change is made at all, a change is made at the node e_1, the upper frontier of X_e. Since this is true throughout the system, the driver knows it. This is enough information to allow him to use a deterministic protocol that essentially guarantees the same payoff as if he had perfect recall, namely 4. If his state is (X_e, α), $\alpha \neq 0$, he should switch to the strategy 0, and perform action B (the only one allowable if he switches to the strategy 0). On the other hand, if his state is $(X_e, 0)$, he should switch to strategy 1 and perform action E (again, the only choice). Unless he starts with initial state $(X_e, 0)$, this protocol is guaranteed to get him home, with payoff 4. Thus, the optimal protocol is better than the optimal strategy in this case.

Clearly what is going on here is that the driver is using the information about his strategy as a device to overcome his memory loss. Is that against the spirit of the absentminded driver paradox? Perhaps so. But notice that something like this *must* be going on in the PR interpretation of the game in Figure 2. If the agent really does switch from strategy f to f' at the node x_1, then the only way that this switch can be realized at the node x_3 is if he can remember that the switch was made. In any case, these examples should serve to point out the

need to make very clear exactly what information the agent has at each node in a game tree, and that this information in general involves far more than just what other nodes he may be at.

5 Discussion

To quote Myerson [1991], "the general form or structure of the models we use to describe games must be carefully considered. A model structure that is too simple may force us to ignore vital aspects of the real games we want to study." As I have tried to argue in this paper, game trees are not always a good tool for representing the information of agents, particularly in games of imperfect recall. It is clear that what an agent does should be a function of what he knows; if an agent cannot distinguish two situations, then he should do the same at both. However, information sets may not adequately characterize what an agent knows. When this is the case, the notion of a strategy as a description of an agent's behavior is inappropriate.

In the systems representation that I have described here, an agent's information is described explicitly at each point in time by his local state, and his behavior is described by means of a protocol, a function a function from local states to actions. The framework forces on the modeler the discipline of making clear exactly what an agent knows at any time. Thus, it allows a modeler to make an independent argument as to when a game tree *is* an adequate representation of an agent's knowledge.

That leads to an obvious question: Just when is the standard game tree model adequate? The pat answer is that it is adequate when the information set is an adequate description of the agent's information. In general, adequacy depends on what we are trying to analyze. Clearly the game tree model is perfectly adequate for many analyses involving games of perfect recall, otherwise it would not have survived so long. On the other hand, PR's examples show that when it comes to analyzing time consistency in games of imperfect recall, information sets are inadequate. The difficulty of using game trees to analyze games of imperfect recall may be one reason that they have received relatively little attention in the game theory literature. (By way of contrast, games of imperfect recall are the norm in the computer science literature. Note that any game played by finite automata that is sufficiently long (in particular, for more steps than the automata have local states), will be a game of imperfect recall, indeed, one that exhibits absentmindedness. Thus, games of imperfect recall arise quite often in practice.) Even in games of perfect recall, game trees cannot be used easily to represent information that players have about other player's strategies. I believe the systems representation could prove useful here as well, especially when it comes to analyzing issues of rationality. I hope to return to this issue in future work.

Although it is beyond the scope of this paper to go into extensive discussion about the systems approach to modeling games, it has some other advantages that are worth pointing out.

- There is no need to assume that agents move sequentially. An "action" may well consist of a tuple of actions, one for each of the agents. (Indeed, this is precisely what is assumed in the more general framework presented in [Fagin, Halpern, Moses, and Vardi 1995].)

For example, if we are considering prisoners' dilemma, where a prisoner may either cooperate (C) or defect (D), and we view these actions as happening simultaneously, then the allowable actions could be (C, C), (C, D), (D, C), or (D, D).

- Using standard semantics of knowledge [Fagin, Halpern, Moses, and Vardi 1995], there is no difficulty in this model of making sense out of one agent knowing that another agent knows something, or there being common knowledge, at any point in the system.

- I have largely ignored the question here of how an agent should assign probabilities to events in systems. This issue is discussed and formalized in some detail in [Halpern and Tuttle 1993], at least for systems that correspond to games with perfect recall. (The philosophical issues of what is an "appropriate" distribution in systems corresponding to games of imperfect recall is also briefly discussed.) In any case, once we associate with each agent a probability distribution that characterizes his beliefs, there is no difficulty in making sense out of statements such as "I know now that if I reach the second intersection, I will place probability $1/10$ on being at the first intersection".

- In the *multi-self* view of decision making [Strotz 1956], an "agent" is viewed as a "team", or a collection of "selves", one associated with each information set. Each "self" makes his choice independently. There is no difficulty capturing the multi-self approach in this framework. Each "self" just becomes another agent.

Of course, given a systems representation of a finite game, it can be viewed as a collection of game trees. By adding a dummy root node, we get a single game tree back. Thus, at some level, it can be argued that the standard game tree model can deal with all the issues I have raised (once we allow an agent's information sets to form a partition of *all* the nodes in a game tree, not just the nodes where the agent moves). I would still argue that it is not always the most convenient way of looking at things, particularly in games of imperfect recall. To back up this argument requires further examples, especially ones showing that the system approach really helps in the analysis of games. This too is something I hope to explore further.

Acknowledgments

I would like to thank Robert Aumann, Paolo Battigalli, Giacomo Bonanno, Ron Fagin, Adam Grove, David Kreps, Bart Lipman, Ariel Rubinstein, and Moshe Vardi for useful discussion and their comments on an earlier version of this paper.

References

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

Battigalli, P. (1995). Dynamic consistency and imperfect recall. Mimeo, Princeton University.

Battigalli, P. and G. Bonanno (to appear, 1995). Synchronic information, knowledge and common knowledge in extensive games. In M. Bacharach, L. A. Gerard-Varet, P. Mongin, and H. Shin (Eds.), *Epistemic Logic and the Theory of Games and Decisions*. Dordrecht, Netherlands: Kluwer.

Fagin, R., J. Y. Halpern, Y. Moses, and M. Y. Vardi (1995). *Reasoning about Knowledge*. Cambridge, Mass.: MIT Press.

Grove, A. J. and J. Y. Halpern (1995). On the expected value of games with absentmindedness. Research Report 9996, IBM.

Halpern, J. Y. (1995). On ambiguities in the interpretation of game trees. Research Report RJ 9995, IBM.

Halpern, J. Y. and R. Fagin (1989). Modelling knowledge and action in distributed systems. *Distributed Computing 3*(4), 159–179. 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".

Halpern, J. Y. and M. R. Tuttle (1993). Knowledge, probability, and adversaries. *Journal of the ACM 40*(4), 917–962. A preliminary version appears in *Proceedings of the 8th ACM Symposium on Principles of Distributed Computing*, 1989, pp. 103–118.

Halpern, J. Y. and M. Y. Vardi (1986). The complexity of reasoning about knowledge and time. In *Proc. 18th ACM Symp. on Theory of Computing*, pp. 304–315.

Hintikka, J. (1962). *Knowledge and Belief*. Ithaca, N.Y.: Cornell University Press.

Myerson, R. B. (1991). *Game Theory*. Cambridge, Mass.: Harvard University Press.

Piccione, M. and A. Rubinstein (1994). On the interpretation of decision problems with imperfect recall. Mimeo (version dated September, 1995).

Rasmusen, E. (1989). *Games and Information: An Introduction to Game Theory*. Oxford, U.K. and Cambridge, Mass.: Basil Blackwell.

Rubinstein, A. (1991). Comments on the interpretation of game theory. *Econometrica 59*, 909–924.

Strotz, R. H. (1956). Myopia and inconsistency in dynamic utility maximization. *Review of Economic Studies 23*, 165–180.