Changing for the better : preference dynamics and agent diversity

Liu, F.

Publication date
2008

Chapter 5

Diversity of Logical Agents in Games

In the preceding chapters, we have seen how agents can have quite diverse preferences, based partly on potentially highly different beliefs. Moreover, they may have different ways of changing these preferences and beliefs. But this is only the beginning of a much longer story of *agent diversity*. It is typical of rational agency that we are *not* all the same, and nevertheless, manage to coordinate activities and exchange information in successful ways. In this chapter, we will look for further clues for this diversity in the setting of concrete activities, viz. *games*. In particular, in games, even before we can get to players' beliefs and preferences, there is the simple basic mechanism of observation of moves that are played, and the information which comes to players because of this. This brings us to sources of diversity having to do with epistemic logic, which will be the main topic pursued here - though we will also provide more material on belief revision in the end as well.

5.1 Introduction: varieties of imperfection

Logical agents are usually taken to be epistemically perfect. But in reality, imperfections are inevitable. Even the most logical reasoners may have limited powers of observation of relevant events, generating uncertainty as time proceeds. In addition, agents can have processing bounds on their knowledge states, say, because of finite memory capacities. This chapter is an exploration of how different types of agents can be described in logical terms, and even co-exist inside the same logical system. Our motivating interest in undertaking this study concerns games with imperfect information, but our only technical results so far concern the introduction of imperfect agents into current logics for information update and belief revision. For a more extensive discussion of issues concerning diversity of agents, we refer to Chapter 6.

5.2 Imperfect information games and dynamic-epistemic logic

Dynamic-epistemic language Games in extensive form are trees $(S, \{R_a\}_{a \in A})$, consisting of nodes for successive states of play, with players' moves represented as binary transition relations between nodes. Imperfect information is encoded by equivalence relations $\sim_i$ between nodes that model uncertainties for player i. Nodes in these structures are naturally described in a combined *modal-epistemic language*. An action modality $[a]\varphi$ is true at a node x when φ holds after every successful execution of move a at x, and a knowledge modality $K_i\varphi$ is true at x when φ holds at every node $y \sim_i x$. As usual, we write $\langle a \rangle$, $\langle K \rangle$ for the existential duals of these modalities. Such a language can describe many common scenarios.

5.2.1. Example. In the following two-step game tree, player ***E*** does not know the initial move that was played by ***A***:

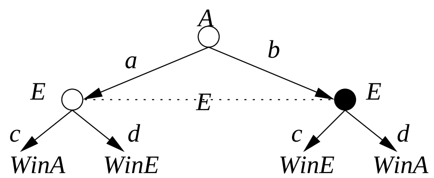

Figure 5.1: Not knowing one's winning move

The modal formula $[a]\langle d \rangle Win_E \wedge [b]\langle c \rangle Win_E$ expresses the fact that ***E*** has a winning strategy in this game, and at the root, she knows both conjuncts. After ***A*** plays move b from the root, however, in the black intermediate node, ***E*** knows merely 'de dicto' that playing either c or d is a winning move, as is expressed by the joint modal-epistemic formula $K_E(\langle c \rangle Win_E \vee \langle d \rangle Win_E)$. But she does not know 'de re' of any specific move that it guarantees a win: $\neg K_E \langle c \rangle Win_E \wedge \neg K_E \langle d \rangle Win_E$ also holds. In contrast, given the absence of dotted lines for ***A***, whatever is true at any stage of this game is known to ***A***. In particular, at the black intermediate node, ***A*** does know that c is a winning move for ***E***.

5.2.2. Remark. (temporal language). For some purposes, it is also useful to have *converse* relations $a^{\cup}$ for moves a, looking back up into the tree. In particular, these help describe play so far by mentioning the moves that have been played, while they also allow us to look back and say what could have happened if play had gone differently. Both are very natural things to say about the course of a game. This is a simple temporal logic variant of the basic modal-epistemic language. For a recent take up in this direction, we refer to [Yap06].

Strategies, plans, and programs A modal-epistemic language describes players' moves and what they know about their step-by-step effects in a game. Explicit information about agents' global behaviour can be formulated in a *dynamic-epistemic language*, which adds complex program expressions. A *strategy* for player i is a function from i's turns x in the game to possible moves at x, while we might think of a *plan* as any relation constraining these choices, though not always to a unique one. Such binary relations and functions can be described using the following expressions

(i) single moves a,

(ii) tests $(\varphi)?$ on the truth of some formula φ,

(iii) use of operations union $\cup$, relational composition ;, and iteration *.

In particular, these operations define the usual slightly more complex program constructs *IF THEN ELSE* and *WHILE DO*. As for test conditions, in this setting, it only makes sense to use φ which an agent *knows to be true or false*. Without loss of generality, we can assume that such conditions have the epistemic form $K_i\varphi$. The resulting programs are called 'knowledge programs' in [FHMV95]. [Ben01] proves that in finite imperfect information games, the following two notions from logic and game theory coincide:

(a) strategies that are defined by knowledge programs,

(b) *uniform strategies*, where players choose the same move at every two nodes which they cannot distinguish.

Valid laws of reasoning about agents and plans Universally valid principles of our language consist of the minimal modal or dynamic logic, and the epistemic logic matching the uncertainty relations – in our case, multi-S5. Logics like this were used in [Moo85] to study planning agents in AI. Of course, here we are most interested in players' changing knowledge as a game proceeds. The language allows us to make these issues more precise. For instance, if a player is certain now that after some move takes place φ is the case, then after that move, is she still certain that φ is the case? In other words, does the following formula hold under all circumstances?

$$K_i[a]p \to [a]K_ip.$$

The answer is negative for most of us. I know that I am boring after drinking – but it does not follow (unfortunately) that after drinking, I know that I am boring. The interchange axiom is only plausible for actions without 'epistemic side-effects'. And the converse implication can be refuted similarly. In general, dynamic-epistemic logic has no significant interaction axioms at all for knowledge and action. If such axioms hold, this must be due to special features of the situation, such as special powers of agents qua observation or memory, or special features of the communicative relationship between agents.

5.2.3. Example. (games versus general dynamic-epistemic models). Imperfect information games themselves do satisfy a special axiom. The tree structure is common knowledge, and players cannot be uncertain about it. This is expressed by the following axiom – where M is the union of all available moves m in the game, and $m^{\cup}$ is the converse relation of m:

$$\langle K\rangle p \rightarrow \langle (M \cup M^{\cup})^{*}\rangle p \qquad (\#)$$

The effect of (#) can be stated as a modal frame correspondence. Epistemically accessible worlds are reachable from the root via sequences of moves:

5.2.4. Fact. (#) is true on a frame iff, for all s, t, if $s \sim_i t$, then $s(M \cup M^{\cup})^{*}t$.

Using this condition, every general model for a modal-epistemic language can be unraveled to a tree of finite action sequences in the usual modal fashion, with uncertainties $\sim_i$ between X, Y just in case $last(X) \sim_i last(Y)$. It is not hard to see that the map from sequences X to worlds $last(X)$ is then a bisimulation for the whole combined language.

Without this constraint, we get 'misty games' ([Höt03]), where players need not know what their moves are or what sort of opponent they are dealing with. This broader setting is quite realistic for planning problems. We return to it at the end of this chapter.

Axioms for perfect agents In the same correspondence style, the above knowledge-action interchange law really describes a special type of agent. To see this, we first observe that

5.2.5. Fact. $K_i[a]p \rightarrow [a]K_i p$ corresponds to the relational frame condition that for all s, t, u, if $sR_a t$ & $t \sim_i u$, then there is a v with $s \sim_i v$ & $vR_a u$.

This condition says that new uncertainties for an agent are always grounded in earlier ones. The equivalence can be proved, e.g. by appealing to the Sahlqvist form of this axiom. Incidentally, this and further observations about the import of axioms may be easier to understand using the equivalent existential versions, here: $\langle a\rangle\langle K\rangle p \rightarrow \langle K\rangle\langle a\rangle p$.

Precisely this relational condition was identified in [Ben01] as a natural version of players having *Perfect Recall* in the game-theoretic sense: They know their own moves and also remember their past uncertainties as they were at each stage. The actual analysis is slightly more complex in the case of games. First, consider nodes where it is the player's turn: then $K_i[a]p$ implies $[a]K_ip$ for the same action a. Perfect Recall does not exclude, however, that moves by one player may be indistinguishable for others, and hence at another player's turn, $K_i[a]p$ implies merely that $[b]K_ip$ for some indistinguishable action b. But there are more versions of perfect recall in game theory. Some allow players uncertainty about the number of moves played by their opponents. [Bon04] has an account of such variants in essentially our correspondence style, now including a temporal operator into the language.

5.2.6. Remark. A similar analysis works for the converse dynamic-epistemic axiom $[a]K_ip \to K_i[a]p$, whose frame truth demands a converse frame condition of 'No Learning', stating essentially that current uncertainty relations remain under identical actions (cf. [FHMV95]). We will encounter this principle in a modified form in Section 5.3.

Agents with Perfect Recall also show special behaviour with respect to their knowledge about complex plans, including their own strategies.

5.2.7. Fact. Agents with Perfect Recall validate all dynamic-epistemic formulas of the form $K_i[\sigma]p \to [\sigma]K_ip$, where σ is a knowledge program.

Proof. By induction on programs. For knowledge tests $(K_i\varphi)?$, we have $K_i[(K_i\varphi)?]p \leftrightarrow K_i(K_i\varphi \to p)$ in dynamic logic, and then $K_i(K_i\varphi \to p) \leftrightarrow (K_i\varphi \to K_ip)$ in epistemic S5, and $(K_i\varphi \to K_ip) \leftrightarrow [(K_i\varphi)?]K_ip$ in dynamic logic. For the program operations of choice and composition, the inductive steps are obvious, and program iteration may be dealt with as repeated composition.□

This simple observation implies that an agent with Perfect Recall who knows what a plan will achieve will also know about these effects halfway through, when some part of his strategy has been played and only some remains. Again, this is not true for all types of agent. This is only one of many delicate issues that can be raised about players' knowledge of their strategies. Indeed, a knowledge statement about *objects*, like 'knowing one's strategy', has aspects that cannot be expressed in our formalism at all. We leave this for further elaboration elsewhere.

Axioms for imperfect agents But there are other types of agents! At the opposite of Perfect Recall, there are agents with bounded memory, who can only remember a fixed number of previous events. Such players with 'bounded rationality' are modelled in game theory by restricting them to strategies that can be implemented by some finite automaton (cf. [OR94]). [Ben01] considers the most drastic form of memory restriction, to just the last event observed. We will call them *memory-free* agents. This kind of agent will be our guiding example of epistemic limitations in this chapter.

In modal-epistemic terms, memory-free agents satisfy a Memory Axiom:

$\langle a \rangle p \rightarrow U[a]\langle K \rangle p$ *MF*

This involves extending our language with a *universal modality* $U\varphi$ stating that φ holds in all worlds. The technical meaning of *MF* is as follows.

5.2.8. Claim. *The axiom MF corresponds to the structural frame condition that, if sR_at and uR_av, then $v \sim_i t$.*

Thus, nodes where the same action has been performed are indistinguishable to memory-free agents. Reformulated in terms of knowledge, the axiom becomes $\langle a \rangle K_i p \rightarrow U[a]p$. This says that the agent can only know things after an action which are true wherever the action has been performed. Therefore, memory-free agents know very little indeed! We will study their behaviour further in Section 5.4. For now, we return to perfection.

5.3 Update for perfect agents

Imperfect information trees merely provide a static record of what uncertainties players are supposed to have at various stages of a game. And then we have to think of some plausible scenario which might have produced these uncertainties. One general mechanism of this kind is provided by *update logics* for actions with epistemic import. Recall Definition 2.5.5 from Chapter 2, where the product rule says that uncertainty among new states can only come from existing uncertainty via indistinguishable actions. That simple mechanism covers surprisingly many forms of epistemic update. [Ben03], [Dit05], [DHK07], [BGP07] and many other recent publications provide introductions to update logics and the many open questions one can ask about them.

The same perspective may now be applied to imperfect information games, where successive levels correspond to successive repetitions of the sequence

$\mathcal{M}$, $\mathcal{M} \times \mathcal{A}$, $(\mathcal{M} \times \mathcal{A}) \times \mathcal{A}$, ...

The result is an obvious tree-like model $\mathit{Tree}(\mathcal{M}, \mathcal{A})$, which may be infinite.

5.3.1. Example. (propagating uncertainty along a game). The following illustration is from [Ben01]. Suppose we are given a game tree with admissible moves (preconditions will be clear immediately). Let the moves come with epistemic uncertainties encoded in an action model, shown in Figure 5.2. Then the imperfect information game can be computed with levels as shown in Figure 5.3:

Now enrich the modal-epistemic language with a dynamic operator

$$\mathcal{M}, s \models \langle \mathcal{A}, a \rangle \varphi \quad \text{iff} \quad (\mathcal{M}, s) \times (\mathcal{A}, a) \models \varphi.$$

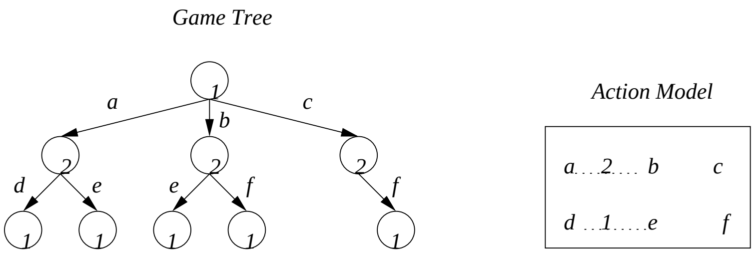

Figure 5.2: Game tree and action model

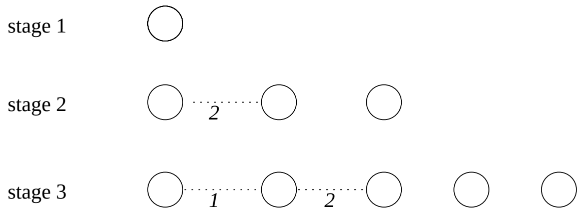

Figure 5.3: propagating uncertainty along a game

Then valid principles express how knowledge is related before and after an action. In particular, we have this key *reduction axiom*:

$$\langle \mathcal{A}, a\rangle\langle K\rangle\varphi \leftrightarrow (PRE(a) \wedge \bigvee\{\langle K\rangle\langle \mathcal{A}, b\rangle\varphi\colon\ a \sim_i b \textit{ for some } b \textit{ in } \mathcal{A}\}).$$

Such laws simplify reasoning about action and planning: We can reduce epistemic properties of later stages to epistemic information about the current stage. From left to right, this axiom is the earlier Perfect Recall, but now with a twist compared with earlier formulations. If an agent cannot distinguish certain actions from the actual one, then those may show up in his epistemic alternatives. The opposite direction from right to left is the No Learning principle. But it does not say that agents can never learn, only that no learning is possible for them among indistinguishable situations by using actions that they cannot distinguish.

The preceding logical observations show that product update is geared toward special agents, viz. those with Perfect Recall. The fact that the reduction axiom is valid shows that perfect memory must have been built into the very definition. And it is easy to see how. The two clauses in defining the new relation $(s,\ a) \sim_i (t,\ b)$ give equal weight to

(a) $s \sim_i t$: past states representing the 'memory component',

(b) $a \sim_i b$: options for the newly observed event.

Changes in this mechanism will produce other 'product agents' by assigning different weights to these two factors (see Section 5.5). But first, we determine the essence of product update from the general perspective of Section 5.2. The following result improves a theorem in [Ben01].

Abstract characterization of product update Consider a tree-like structure $\mathcal{E}$ with possible events (or actions) and uncertainty relations among its nodes, which can also verify atomic propositions p, q, ... The only contrast with a real tree is that we allow a bottom level with multiple roots. Nodes X, Y, ... are at the same time finite sequences of events, and the symbol $^\cap$ expresses concatenation of events. Intuitively, we think of such a tree structure $\mathcal{E}$ as the possible evolutions of some process – for instance, a game. A particular case is the above model *Tree*$(\mathcal{M}, \mathcal{A})$ starting from an initial epistemic model $\mathcal{M}$ and an action model $\mathcal{A}$, and repeating product updates forever. Now, the preceding discussion shows that the following two principles are valid in *Tree*$(\mathcal{M}, \mathcal{A})$, which can be stated as general properties of a tree $\mathcal{E}$. They represent Perfect Recall and 'Uniform No Learning', respectively:

PR If $X^\cap(a) \sim_i Y$, then $\exists b \exists Z : Y = Z^\cap(b)$ & $X \sim_i Z$.

UNL If $X^\cap(a) \sim_i Y^\cap(b)$, then $\forall U, V : if\ U \sim_i V$, then $U^\cap(a) \sim_i V^\cap(b)$, provided that $U^\cap(a), V^\cap(b)$ both occur in the tree $\mathcal{E}$.

Moreover, the special nature of the preconditions in product update, as definable conditions inside the current epistemic model, validates one more abstract constraint on the tree $\mathcal{E}$:

BIS-INV The set $\{X \mid X^\cap(a) \in \mathcal{E}\}$ of nodes where action a can be performed is closed *under purely epistemic bisimulations* of nodes.

Now we have all we need to prove a converse representation result.

5.3.2. Theorem. *For any tree $\mathcal{E}$, the following are equivalent:*

(*a*) $\mathcal{E} \cong Tree(\mathcal{M}, \mathcal{A})$ *for some* $\mathcal{M}, \mathcal{A}$.

(*b*) $\mathcal{E}$ *satisfies PR, UNL, BIS-INV.*

Proof. From (a) to (b) is the above observation. Now, from (b) to (a). Define an epistemic model $\mathcal{M}$ as the set of initial points in $\mathcal{E}$ and copy the relations $\sim_i$ from $\mathcal{E}$. The action model $\mathcal{A}$ contains all possible actions occurring in the tree, where we set

$$a \sim_i b \quad \text{iff} \quad \exists X \exists Y : X^\cap(a) \sim_i Y^\cap(b).$$

We also need to know that the *preconditions* $PRE(a)$ for actions a are as required. For this, we use the well-known fact that in any epistemic model, any set of worlds that is closed under epistemic bisimulations must have a definition in the epistemic language – though admittedly, one allowing infinite conjunctions and disjunctions. The abstract setting of our result allows no further finitization of this definability.

Now, the obvious identity map F sends nodes X of $\mathcal{E}$ to corresponding states in the model *Tree* $(\mathcal{M}, \mathcal{A})$. First, we observe the following fact about $\mathcal{E}$ itself:

5.3.3. Lemma. *If $X \sim_i Y$, then length(X) = length(Y).*

Proof. If X, Y are initial points in $\mathcal{E}$, both their lengths are 0. Otherwise, suppose X has length n+1. By *PR*, X's initial segment of length n stands in the relation $\sim_i$ to a proper initial segment of Y whose length is that of Y minus 1. Repeating this observation peels off both sequences to initial points after the same number of steps. □

5.3.4. Claim. *$X \sim_i Y$ holds in $\mathcal{E}$ iff $F(X) \sim_i F(Y)$ holds in $Tree(\mathcal{M}, \mathcal{A})$.*

The proof is by induction on the common length of the two sequences X, Y. The case of initial points is clear by the definition of $\mathcal{M}$. As for the inductive steps, consider first the direction $\Rightarrow$. If $U^\frown(a) \sim_i V$, then by *PR*, $\exists b \exists Z : V = Z^\frown(b)$ & $U \sim_i Z$. By the inductive hypothesis, we have $F(U) \sim_i F(Z)$. We also have $a \sim_i b$ by the definition of $\mathcal{A}$. Moreover, given that the sequences $U^\frown(a)$, $Z^\frown(b)$ both belong to $\mathcal{E}$, their preconditions as listed in $\mathcal{A}$ are satisfied. Therefore, in $Tree(\mathcal{M}, \mathcal{A})$, by the definition of product update, $(F(U), a) \sim_i (F(Z), b)$, i.e. $F(U^\frown(a)) \sim_i F(Z^\frown(b))$.

As for the direction $\Leftarrow$, suppose that in $Tree(\mathcal{M}, \mathcal{A})$ we have $(F(U), a) \sim_i F(Z), b)$. Then by the definition of product update, $F(U) \sim_i F(Z)$ and $a \sim_i b$. By the inductive hypothesis, from $F(U) \sim_i F(Z)$ we get $U \sim_i Z$ in $\mathcal{E}(*)$. Also, by the given definition of $a \sim_i b$ in the action model $\mathcal{A}$, we have $\exists X \exists Y$: $X^\frown(a) \sim_i Y^\frown(b)(**)$. Taking $(*)$ and $(**)$ together, by *UNL* we get $U^\frown(a) \sim_i Z^\frown(b)$, provided that $U^\frown(a)$, $V^\frown(b) \in \mathcal{E}$. But this is so since the preconditions $PRE(a)$, $PRE(b)$ of the actions a, b were satisfied at $F(U), F(Z)$. This means these epistemic formulas must also have been true at U, V – so, given what $PRE(a)$, $PRE(b)$ defined, $U^\frown(a)$, $V^\frown(b)$ exist in the tree $\mathcal{E}$. □

This result is only one of a kind, and its assumptions may be overly restrictive. In many game scenarios, preconditions for actions are not purely epistemic, but rather depend on what happens over time. E.g. a game may have initial factual announcements – like the Father's saying that at least one child is dirty in the puzzle of the Muddy Children. These are not repeated, even though their preconditions still hold at later stages. Describing this requires preconditions $PRE(a)$ for actions a that refer to the temporal structure of the tree $\mathcal{E}$, and then the above invariance for purely epistemic bisimulations would fail. Another strong assumption is our use of a single action model $\mathcal{A}$ that gets repeated all the time in levels $\mathcal{M}$, $(\mathcal{M} \times \mathcal{A})$, $(\mathcal{M} \times \mathcal{A}) \times \mathcal{A}$, ... to produce the structure $Tree(\mathcal{M}, \mathcal{A})$. A more local perspective would allow different action models $\mathcal{A}_1$, $\mathcal{A}_2$, ... in stepping from one tree level to another. And an even more finely-grained view arises if single moves in a game themselves can be complex action models. In the rest of this paper, for convenience, we stick to the single-model view.

5.4 Update logic for bounded agents

Limitations on information processing The information-processing capacity of agents may be bounded in various ways. One of these is 'external': Agents may have restricted powers of observation. This kind of restriction is built into the definition of action models, with uncertainties for agents – and the product update mechanism of Section 5.3 reflects this. Another type of restriction is 'internal': Agents may have bounded memory. Agents with Perfect Recall had limited powers of observation but perfect memory. At the opposite extreme we find memory-free agents who can only observe the last event, without maintaining any record of what went on before. In this section, we explore this extreme case.

Characterizing types of agent In the preceding, agents with Perfect Recall have been described in various ways. Our general setting was the tree $\mathcal{E}$ of event sequences, where different types of agents i correspond to different types of uncertainty relation $\sim_i$. One approach was via *structural conditions* on such relations, such as *PR*, *UNL*, and *BIS-INV* in the above characterization theorem. Essentially, these three constraints say that

$$X \sim_i Y \quad \text{iff} \quad length(X) = length(Y) \text{ and } X(s) \sim_i Y(s) \text{ for all positions } s.$$

Next, these conditions also validated corresponding *axioms in the dynamic-epistemic language* that govern typical reasoning about the relevant type of agent. But thirdly, we can also think of agents as a sort of *processing mechanism*. Intuitively, an agent with Perfect Recall is a push-down store automaton maintaining a stack of all past events and continually adding new observations to the stack. Such a processing mechanism was provided by our representation theorem, viz. epistemic product update.

Bounded memory Another broad class of agents arises by assuming bounded memory up to some fixed finite number k of positions. In general trees $\mathcal{E}$, this makes two event sequences X, Y $\sim_i$-equivalent for such agents i iff their last k positions are $\sim_i$-equivalent. In this section we only consider the most extreme case of this, viz. *memory-free agents* i:

$$X \sim_i Y \quad \text{iff} \quad last(X) \sim_i last(Y) \text{ or } X = Y = \text{the empty sequence} \qquad (\$)$$

Agents of this sort only respond to the last-observed event. In particular, their uncertainty relations can now cross between different levels of a game tree: They need not know how many moves have been played. Perhaps contrary to appearances, such limited agents can be quite useful. Examples are *Tit-for-Tat* players in the iterated Prisoner's Dilemma which merely repeat their opponents' last move ([Axe84]), or *Copy-Cat* players in game semantics for linear logic which can win 'parallel disjunctions' of games $G \vee G^d$ ([Abr96]). Incidentally, these are players with a hard-wired *strategy*: a point that we will discuss below. It is easy to characterize such agents in terms similar to what we did with Perfect Recall.

5.4.1. Fact. An equivalence relation $\sim_i$ on $\mathcal{E}$ is memory-free in the sense of ($) if and only if the following two conditions are satisfied:

PR^- If $X^\frown(a) \sim_i Y$, then $\exists\, b \sim_i a \exists\, Z : Y = Z^\frown(b)$.
UNL^+ If $X^\frown(a) \sim_i Y^\frown(b)$, then $\forall\, U, V : U^\frown(a) \sim_i V^\frown(b)$, provided that $U^\frown(a)$, $V^\frown(b)$ both occur in the tree $\mathcal{E}$.

Proof. If an agent i is memory-free, its relation $\sim_i$ evidently satisfies PR^- and UNL^+. Conversely, suppose that these conditions hold. If $X \sim_i Y$, then either X, Y are both the empty sequence, and we are done, or, say, $X = Z(a)$. Then by PR^-, $Y = U(b)$ for some $b \sim_i a$, and so $last(X) \sim_i last(Y)$. Conversely, the reflexivity of $\sim_i$ plus UNL^+ imply that, if the right-hand side of the equivalence ($) holds, then $X \sim_i Y$. □

It is also easy to give a characteristic modal-epistemic axiom for this case. First, we set the following

$$a \sim_i b \quad \text{iff} \quad \exists X \exists Y : X^\frown(a) \sim_i Y^\frown(b).$$

5.4.2. Fact. The following equivalence is valid for memory-free agents:

$$\langle a \rangle \langle K \rangle \varphi \leftrightarrow (PRE(a) \& E \bigvee_{b \sim_i a} \langle b \rangle \varphi).$$

Here $E\varphi$ is an additional *existential modality* saying that φ holds in at least one node. This axiom looks at first glance like the Perfect Recall axiom of Section 3, but note that there is no epistemic modality $\langle K \rangle$ on the right-hand side of the equivalence. Also, this new axiom implies axiom *MF* from Section 5.2, assuming that basic actions are partial functions.

5.4.3. Remark. (reduction axioms for an existential modality). Once the static description language gets extended, to restore the harmony of an update logic, one should also extend the dynamic update reduction axioms with a clause for the new operator. E.g., returning to Section 5.3, the following reduction axiom is valid for standard product update:

$$\langle \mathcal{A}, a \rangle E\varphi \leftrightarrow (PRE(a) \wedge E \bigvee \langle \mathcal{A},\, b \rangle \varphi \textit{ for some } b \textit{ in } \mathcal{A}).$$

The process mechanism: finite automata The processor of memory-free agents is a very simple *finite automaton* creating their correct $\sim_i$ links:

> States of the automaton: all equivalence classes $X^{\sim_i}$
> Transitions for actions a: $X^{\sim_i}$ goes to $(X^{\frown}(a))^{\sim_i}$

There are only finitely many states since we had only finitely many actions in the game tree $\mathcal{E}$. The transitions are well-defined, since by the No Learning assumption UNL^+, if $X \sim_i Y$, then $X^{\frown}(a) \sim_i Y^{\frown}(a)$. The automaton starts in the equivalence class of the empty event sequence. Repeating transitions, it is easy to see that

> When the automaton is given the successive members of an event sequence X as input, it ends in state $X^{\sim_i}$.

In particular, $X \sim_i Y$ iff the automaton ends in the same state on both of these event sequences. Moreover, the combination of the conditions UNL^+ and PR^- on memory-free agents tells us something about the special type of automaton that suffices:

> All transitions a end in the same state (as $X^{\frown}(a) \sim_i Y^{\frown}(a)$ for all X, Y), and by PR^-, no transition ends in the initial state.

Let us call such automata *rigid*. They only have states for the last-observed event, and such states will even coincide when the events are not epistemically distinguishable for the agent.

5.4.4. FACT. Memory-free agents are exactly those whose uncertainty relation is generated by a rigid finite-state automaton.

Of course, more complex finite automata can have more differentiated responses to observed events a, up to some fixed finite number of cases.

5.4.5. REMARK. (automata theory). Connections with automata theory, in particular the Nerode representation of finite automata recognizing regular sets of event sequences, are found in [BC03]. The above framework can be extended with more general preconditions for game actions referring to time, by generalizing to the action/test automata used for propositional dynamic logic in [HKT00].

Strategies and automata The preceding automata for bounded agents are reaction devices to incoming observations. But it is also tempting to think of automata as generators of behaviour – in particular, as specific *strategies*. The latter view is more in line with the usual treatment of our motivating examples, like *Tit-for-Tat* or *Copy-Cat*. A strategy for player i in a game is a function assigning moves to turns for i, these moves are responses to *other players'* actions.

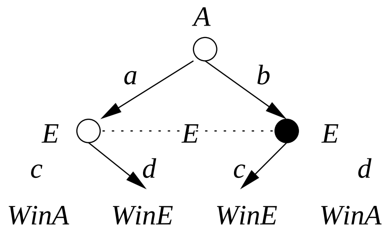

Figure 5.4: Winning strategy

This is easily visualized in game trees $\mathcal{E}$. E.g., player $\boldsymbol{E}$'s winning strategy in the game of Section 5.2 looks as shown in Figure 5.4.

But the reflection in finite automata will be a little different then, as players do not respond to a last action if played by themselves (these are 'non-events' for the purpose of a strategy). Thus, the usual automaton for *Tit-for-Tat* encodes actions by the agent itself as *states*, while actions by the opponent are the true observed events, shown in Figure 5.5.

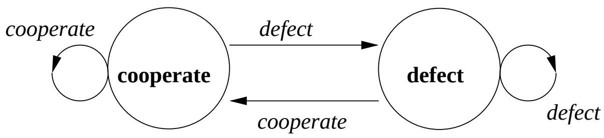

Figure 5.5: Tit for Tat

We do not undertake an integration of the two sorts of finite automata here. Either way, the simplicity of such automata for agents and their strategies may also be seen by considering the special syntactic form of memory-free strategies as simple knowledge programs in the dynamic-epistemic language.

This concludes our discussion of memory-free agents per se. To highlight them even more, we add a few contrasts with agents with Perfect Recall.

Differences in what agents know Memory-free agents i know less than agents with Perfect Recall. The reason is that their equivalence classes for $\sim_i$ tend to be larger. E.g., *Tit-for-Tat* only knows she is in two of the four possible matrix squares (*cooperate, cooperate*) or (*defect, defect*). But amongst many other failures, she does not know the accumulated score at the current stage. It is also tempting to say that memory-free agents can only run very simplistic strategies. But this is not quite right, since any knowledge program makes sense for all agents. The point is just that certain knowledge conditions will evaluate differently for both. E.g., a Perfect Recall agent may be able to act on conditions like "action a has occurred twice so far", which a memory-free agent can never execute, since she can never know that the condition holds. Thus the difference is rather in the number of non-equivalent available uniform strategies and the successful behaviour guaranteed by these.

5.4.6. Example. Consider the following game tree for an agent ***A*** with perfect information, and a memory-free agent ***E*** who only observes the last move.

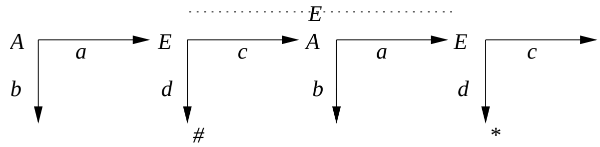

Figure 5.6: How memory-free agents may suffer

Suppose that outcome # is a bad thing, and $*$ a good thing for ***E***. Then the desirable strategy "play d only after you have seen two a's" is unavailable to ***E*** – while it is available to a player with Perfect Recall.

Another difference between Perfect Recall agents and memory-free agents has to do with what they know about their *strategies*. We saw that an agent with Perfect Recall for atomic actions also satisfies the key implication

$K_i[\sigma]p \to [\sigma]K_ip$, when σ is any complex knowledge program.

By contrast, the *MF* Memory Axiom

$$\langle a\rangle p \to U[a]\langle K\rangle p.$$

does not 'lift' to arbitrary knowledge programs instead of the single action a. To see this, it suffices to look at the case of a choice program $a \cup b$. Our eventual reduction version

$$\langle a\rangle\langle K\rangle\varphi \leftrightarrow (PRE(a)\&E \bigvee_{b\sim_i a} \langle b\rangle\varphi).$$

is a bit harder to generalize, because we would first have to analyze what it means to be indistinguishable from a complex action.

Memory and time A good way of making differences between agents more explicit is the introduction of a richer language. So far, we have mostly looked at a purely epistemic language for preconditions and an epistemic language with forward action modalities for describing updates or general moves through a game tree. With such a language, some of the intuitive distinctions that we want to make between different agents cannot be expressed. E.g., suppose that there is just one initial world s and one action, the identity *Id*, which always succeeds:

$$s \qquad (s, Id) \qquad ((s, Id), Id) \qquad ...$$

Thus, each horizontal level contains just one world. In this model, the uncertainty lines for Perfect Recall agents and memory-free agents are different. The latter see all worlds ending in *Id* as indistinguishable, whereas product update for the former makes all worlds different. Nevertheless, agents know exactly the same purely epistemic statements in each world. The technical reason is that all states are epistemically bisimilar, and composing the uncertainty lines for a player with bisimulation links makes no difference to what she knows. But intuitively, the Perfect Recall player should know how many actions have occurred, since her uncertainties did not cross levels. Now, if we want to let agents know explicit statements about where they are in the game, we can add the backward-looking *converse action modalities* mentioned in Section 5.2. Then an agent knows, e.g., that two moves have been played if she knows that two consecutive converse actions are possible, but not three. Thus, a temporal dynamic-epistemic language is more true to what we would want to say intuitively about players and their differences. Moreover, this language can also express more complex preconditions for actions, resulting in the definability of a much broader range of strategies (cf. [Rod01], [BP06] and [BGP07]).

5.4.7. Remark. (backward-looking update). A backward-looking temporal language also enriches update logic. Our reduction axioms so far were forward-looking analysis of *preconditions*, reducing what agents know after an action has taken place to what they knew before. What about converse reduction axioms of the following form, say:

$$\langle a^{\cup}\rangle\langle i\rangle\varphi \leftrightarrow (PRE_{a^{\cup}} \& E \bigvee_{b^{\cup}\sim_i a^{\cup}} \langle b^{\cup}\rangle\varphi)?$$

These are related to *postconditions* for actions a: The strongest that we can say when a was performed in a world satisfying φ is that $\langle a^{\cup}\rangle\varphi$ must hold. Such postconditions are known to be impossible to define, even for simple public announcements, in the open-ended total universe of all epistemic models. But things are more controlled in our trees $\mathcal{E}$ which fix the previous history for any current world. In that case, we can convert at least earlier full commutativity axioms like the interchange of $\langle a\rangle\langle K\rangle$ and $\langle K\rangle\langle a\rangle$ to backward-looking versions. For more discussions, again we refer to [Yap06].

A final caveat This discussion has been somewhat impressionistic. In particular, it is easy to *over-interpret* our formal models in terms of 'knowledge talk'. At any given state, the bare fact is that an agent i has the set of all its $\sim_i$ alternatives. Depending on how *we* describe that set, we attribute various forms of knowledge to the agent. But most of these are just correlations – like when we say that *Tit-for-Tat* knows that it is in a 'cooperative' state. Such a description need not correspond to any *representational attitude* inside the agent. This mismatch is a limitation of epistemic logic in general, and over-interpretation occurs just as well for agents with Perfect Recall. These are triggered by possibly complex 'horizontal' knowledge conditions $K\varphi$ referring to the current tree level in structures like $\mathcal{E}$ or $Tree(\mathcal{M}, \mathcal{A})$. But we, as outside observers, may identify these as equivalent to simple assertions about the past of the process, such as "action a has occurred twice". And even when we use the above richer temporal language, this still need not imply matching richer representations inside the agent.

5.5 Creating spectra of agents by modulating product update rules

Toward a spectrum of options Perfect Recall agents and memory-free agents are two extremes with room in the middle. Using the automata of Section 5.4, one might define update for progressively better informed k-bit agents having k memory cells, creating much great diversity. By contrast, agents with Perfect Recall seemed the natural children of product update. But even here there is room for alternative stipulations! The following type of agent is closely related to the memory-free ones discussed before.

Forgetful updaters As we saw in Section 5.3, product update for new uncertainties mixed a memory factor (viz. uncertainty between old states) and an observation factor (viz. uncertainty between actions). Agents might weigh these differently. A memory-free agent, by necessity, gives weight 0 to the past. If updating agents only remember their last action, how do they update their information? Here is a simple new definition. We drop the memory factor when defining product models $\mathcal{M} \times \mathcal{A}$, and set:

$$(x, a) \sim_i (y, b) \quad \text{iff} \quad a \sim_i b!$$

Thus, new uncertainty comes only from uncertainty about observed actions. Just as before, this leads to a valid *reduction axiom*:

5.5.1. Fact. The following equivalence is valid with forgetful update:

$$\langle \mathcal{A}, a \rangle \langle K \rangle \varphi \leftrightarrow (PRE(a) \wedge E \bigvee \langle \mathcal{A}, b \rangle \varphi\text{: } a \sim_i b \text{ for some } b \text{ in } \mathcal{A}).$$

As before, to restore the harmony of the complete system, we also need a reduction axiom for the new modality E, which turns out to be

$$\langle \mathcal{A}, a \rangle E\varphi \leftrightarrow (PRE(a) \wedge E \bigvee \langle A,\ b \rangle \varphi \text{ for some } b \text{ in } \mathcal{A}).$$

And it is also possible to give an abstract characterization of forgetful updaters by modifying the main theorem of Section 5.3.

In the original version of this chapter, it was suggested that forgetful updaters are precisely the memory-free agents of Section 5.4. But as was pointed out by Josh Snyder (personal communication), this seems wrong. Consider the following scenario. A forgetful updater is uncertain between world s with p and world t with $\neg p$. There are two possible actions:

- a with precondition: $p \wedge \neg Kp$,
- b with precondition: $Kp \vee (\neg p \wedge \neg K \neg p)$.

Let the actual actions be a, b in that order. Then the successive product updates for forgetful updaters are

(i) from $\{s, t\}$ to $\{(s, a), (t, b)\}$, without an uncertainty link, so the agent knows that p in the actual world (s, a), whereas he knows that $\neg p$ in the unrelated world (t, b)

(ii) from $\{(s, a), (t, b)\}$ to $\{((s, a), b)\}$, since neither a nor b can be performed in (t, b).

But in that final model, the agent still knows that p, even though a memory-free agent would not know p because she would be uncertain between $((s, a), b)$ and (t, b). [Sny04] has a solution for this by modifying product update so as to keep all worlds around, whether or not preconditions of actions are satisfied, while redefining uncertainty relations in some appropriate fashion. Another option may be the addition of suitable 'copy actions' that keep earlier sequences alive at later levels. We will come back to these two proposals in Chapter 6.

The upshot of this discussion is that forgetful updaters are not the same as our earlier memory-free agents, although they are close. In the remainder of this section, we mention some other modulations on product update that create different types of agents.

Probabilistic modulations Letting agents give different weights to memory and observation in computing a new information state is an idea from a well-known tradition preceding modern update logics, viz. inductive logic and Bayesian statistics. Different agents or 'inductive methods' differ in the weight they put on experience versus observation. To implement this perspective in update logics, we need a *probabilistic* version of product update, as first defined in [Ben03], and later developed in [BGK06].

Belief revision and plausibility update But staying closer to our qualitative setting, we can also give another natural example of diversity with a numerical flavour. In the theory of *belief revision*, it has long been recognized that agents may obey different rules, more conservative or more radical, when incorporating new information. Such rules are different options for computing new states on the basis of incoming evidence. Such diversity will even arise for agents with epistemic Perfect Recall, as we will now show.

In general, information update is a different mechanism from belief revision, but the two viewpoints can be merged. [Auc03] adds a function κ to epistemic models $\mathcal{M}$ and action models $\mathcal{A}$ which assigns *plausibility values* to states and actions. Here $\kappa_i(v) > \kappa_i(w)$ means that agent i believes that world w is more plausible than world v. This allows us to define degrees of belief in a proposition as truth in all worlds up to a certain plausibility:

$$\mathcal{M},\, s \models B_i^\alpha \varphi \text{ iff } \mathcal{M},\, t \models \varphi \text{ for all worlds } t \sim_i s \text{ with } \kappa(t) \leq \alpha.$$

Incidentally, we can also define $B_i^\alpha \varphi$ as $K_i(\kappa_i^\alpha \rightarrow \varphi)$, provided we add suitable propositional constants κ_i^α to the language (cf. [Liu04]).

Next, plausibilities of actions indicate what an agent believes about what most likely took place. Computing the plausibility of a new state $(w,\, a)$ in a product model $\mathcal{M} \times \mathcal{A}$ requires some intuitive rule. Aucher himself proposes an 'addition formula' for κ-values, subtracting a 'correction factor':

$$\kappa_j'(w, a) = Cut_M(\kappa_j(w) + \kappa_j^*(a) - \kappa_j^w(PRE(a))).$$

Here *Cut* is a technical 'rescaling' device, and the correction $\kappa_j^w(PRE(a))$ is the smallest κ-value in $\mathcal{M}$ among all worlds $v \sim_i w$ satisfying $PRE(a)$.

A continuum of revision rules In our current perspective, we see this stipulation not as the unique update rule for plausibility but as a choice for a particular type of agent. Aucher's formula makes an agent 'eager' in the following sense: The factor for the last-observed action weighs just as heavily as that for the previous state, even though the latter might encode a long history of earlier beliefs. But we can easily create further diversity by changing the above formula into one with parameters λ and μ:

$$\kappa_j'(w, a) = \frac{1}{\lambda + \mu}(\lambda \kappa_j(w) + \mu \kappa_j^*(a)).$$

By changing values of λ and μ, we can distinguish many different types of agents. Diversity increases even further when we let agents assign different plausibility values to preconditions of actions. For a detailed discussion, see [Liu04].

5.5.2. Remark. (belief revision by bounded agents). It is also possible to use ideas from Section 5.3, and consider belief revising agents with bounded memory. For a more extensive study of belief revision by agents with bounded resources, we refer to [Was00], [ALW04], and [AJL07].

Coming to terms with belief revision, in addition to information update, is natural – also from our motivating viewpoint of games. After all, players of a game surely do not just update on the basis of observed past moves. They also revise their expectations about future actions of opponents. Further examples of this will arise in our final sections.

5.6 Mixing different types of agents

So far, we have looked at agent types separately. But agents live in groups, whose members may have different types. Turing machines might communicate with finite automata, and humans occasionally meet Turing machines, like their computers, or finite automata, like very stupid people. What makes groups of agents most interesting is that they *interact*. In this setting, a host of new questions arises – of which we discuss just a few.

Uncertainty and exploitation Do different types of agents know each other's type? There is an issue of definition first. What does it *mean* to know the type of another agent? One could think of this, e.g., as knowing that the agent satisfies all axioms for its type, as formulated in Sections 5.2, 5.3 and 5.4. But then, in imperfect information games, or the more general trees $\mathcal{E}$ studied above, the types of all agents are *common knowledge*, because these axioms hold everywhere in the tree. Introducing ignorance of types requires more complex structures in the sense of [Höt03]. Suppose that agent $\boldsymbol{A}$ does not know if his opponent is a memory-free agent or not. Then we need disjoint unions of game trees with uncertainty links between them. Indeed, this extension already arises when we assume that some agent i does not know the precise uncertainties of its opponent between i's actions. Consider the following example:

5.6.1. Example. The following situation is a simple variant of Example 5.1, pictured in Figure 5.7.

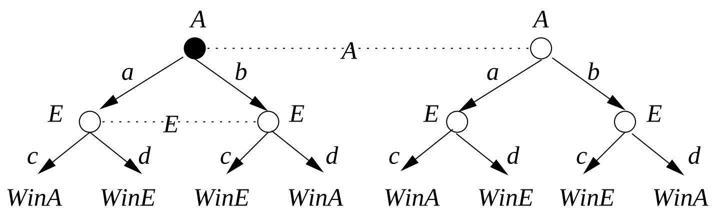

Figure 5.7: Ignorance of the opponent type

Right at the start of the game, agent $\boldsymbol{A}$ does not know whether $\boldsymbol{E}$ has limited powers of observation or not. In particular, note that the earlier axiom $\langle K\rangle p \to \langle (M \cup M^{\cup})^*\rangle p$ for imperfect information games fails here. The 'second root' toward the right is an epistemic alternative for $\boldsymbol{A}$, but it is not reachable by any sequence of moves.

Can an agent take advantage of knowing another agent's type? Of course. It would be tedious to give overly formal examples of this, since we all know this phenomenon in practice. Suppose that I know that after returning a serve of mine, you always step toward the middle of the court. Then passing you all the time along the outer line is a simple winning strategy. A more dramatic scenario of this sort occurs in the movie "Memento" about a man who has lost his long-term memory and has fallen into the hands of unscrupulous cops and women. But *must* a memory-free agent do badly against a more sophisticated epistemic agent? That depends on the setting. E.g., memory-free *Tit-for-Tat* managed to win against much more sophisticated computer programs ([Axe84]). But even this does not do justice to the complexity of interaction!

Learning and revision over time In practice, we may not know the types of other agents and may need to *learn* them. Such learning mechanisms are themselves a further source of interesting epistemic diversity, as is pointed out in [Hen01] and [Hen03]. In general, there is no guarantee at all that a learning method will reveal the type of an opponent. Evidently, observing a finite number of moves can never tell us for certain whether we are playing against an agent with Perfect Recall or against a finite-state automaton with a large finite memory beyond the current number of rounds played so far. But there is a weaker sense of learning that may be more relevant here. We may enter a game with certain hypotheses about the agents that we are playing against. And such hypotheses can be updated by observations that we make as time goes by. E.g. I can *refute* the hypothesis that you are a memory-free agent by observing different responses to the same move of mine at different stages of the game. Or, I can have the justified hypothesis that you are memory-free, and one observed response to a move of mine then reveals a part of your fixed strategy.

Two kinds of update Intuitively, the game situations just described go beyond the information and plausibility update of Sections 5.3, 5.4, 5.5. But to arrive at a more definite verdict, one has to separate concerns. The above questions involve many general issues about update that arise even without diversity of agents. For instance, learning about one's opponent's type is akin to the well-known question of learning one's opponent's strategy. Types may be viewed as sets of strategies, so learning the type amounts to some useful intermediate reduction in the strategic form of the game. In what follows, we will illustrate a few issues in a concrete scenario.

5.6.2. Example. Consider the following game of perfect information. Suppose that $\boldsymbol{A}$ knows that $\boldsymbol{E}$ is memory-free: What does it take him then to find out which particular strategy $\boldsymbol{E}$ is running? See Figure 5.8.

This scenario illustrates the danger in discussing these matters. For, if $\boldsymbol{A}$ *knows* that $\boldsymbol{E}$ is memory-free, the latter fact is true, and hence, at her second turn, $\boldsymbol{E}$ can never play d, since she has already played c in response to b in order to get there at all. So, we can only sensibly talk about *beliefs* here. In the simplest case, these can be modelled as

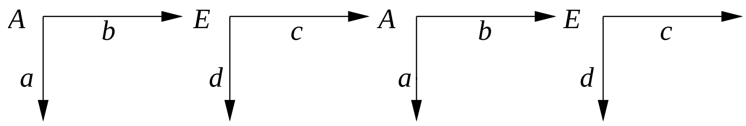

Figure 5.8: Finding out about types and strategies

> *subsets of all runs of the game from now on*,

viz. those future runs which the agent takes to be most likely. Thus, ***A***'s belief would rule out the 'non-homogeneous run' for ***E*** in this game, even though further observation might refute the belief, forcing ***A*** to revise. Now, *belief revision* means that, as the game is played and moves are observed, this set of most plausible runs gets modified. E.g., suppose that ***E*** in fact plays d at her first turn. Then the hypothesis that she was memory-free seems vindicated, and we also know part of her strategy. But this is again too hasty. We have not tested any global assertion about her strategy, precisely because the game is over, and we have no means of observing what ***E*** would have done at her second turn.

Thus, we must be sensitive to distinctions like 'predicting what *will* happen' versus 'predicting what *would* happen' in some stronger counterfactual sense. Hypotheses about one's opponents' type are of the latter sort, and they may be harder to test. The representation of alternative scenarios and suitable update mechanisms over these need not be the same in both cases. In particular, we might need two kinds of update mechanisms. One is the local computation of players' uncertainties at nodes of the game concerning facts and other players' information, as described by the earlier product update and plausibility update. The other is the changing of the global longer-term expectations about strategies over time by observing the course of the game.

5.6.3. Remark. (local versus global update?). Despite the appealing distinction made just now, uncertainty about the future can sometimes be 'folded back' into local update. Consider any game of perfect information. Uncertainties about the strategy played by one's opponent may be represented in a new *imperfect information* game, whose initial state consists of all possible strategy profiles with appropriate uncertainty lines for players between these. Update on such a structure occurs as consecutive moves are played in the game, which can be seen as a form of public announcement ruling out certain profiles from the diagram. Likewise, belief revision becomes plausibility update on strategy profiles. For details, see [Ben04a] and [Ben04b].

Update can get even more subtle than this with learning global types. Consider the earlier Example 5.7 where ***A*** did not know if ***E*** had perfect information or not. How can ***A*** find out? If only moves are observed, we would have to say that having just a single uncertainty line for $\boldsymbol{A}$ between the real root and the 'pseudo-root' makes no sense. For, after move a is played, $\boldsymbol{A}$ has learnt nothing that would now enlighten him, so there should be an uncertainty line at the mid-level as well. But in another sense, $\boldsymbol{A}$ *has* learnt something! He now knows that $\boldsymbol{E}$ is uncertain, so he is in the game on the left. To make sense of this second scenario, we have to assume that introspection into $\boldsymbol{A}$'s epistemic state also counts as an update signal.

We leave matters here. What we hope to have shown is that diversity of agents raises some interesting issues, while sharpening our intuitions about the required mix of update and revision in games. In particular, instead of theorizing about abstract revision mechanisms, a hierarchy of agent types suggests very concrete switching scenarios as our beliefs about a type get contradicted by events in the course of the game.

Merging update logic and temporal logic To make sense of the issues in this section, we need to introduce a richer framework than our dynamic-epistemic logic so far. We now need to maintain global hypotheses about behaviour of agents in future courses of the game, which can be updated as time proceeds. This temporal intuition reflects computational practice, as well as philosophical studies of agency and planning (cf. [BPX01]). It is also much like questions in standard game theory about predicting the future behaviour of one's opponents: 'rational', or less so. Technically, we think the best extension for this broader sort of update would be *branching temporal models* with a suitable language referring to behaviour over time (cf. [FHMV95], [PR03]). The above tree structures $\mathcal{E}$ can easily support such a richer language. [Ben04b] has a few speculations on update in such a temporal setting, and a recent exploration can be found in [BP06].

5.7 Conclusion

The point of this chapter is that diversity of agents is a fact of life, and moreover, that it is interesting from a logical point of view. Indeed, as we shall see in Chapter 6, one can even apply it to other logical core tasks, such as inference by more clever and more stupid agents. Technically, we have shown that it is easy to describe different kinds of epistemic agents in dynamic epistemic logics, and that this style of analysis matches well with information flow in extensive games of imperfect information.

Several interesting further questions arise now, and some of them have been taken up in the time since the paper [BL04] behind this chapter was first published. One line of extension is the further mathematical study of special patterns in arbitrary imperfect information games, viewed as trees of actions with epistemic uncertainties. In such a setting, our representation results may have more sophisticated versions for other kinds of behaviour. One could see this as pursuing the fine-structure of general models for dynamic-epistemic logic. Indeed, some generalized versions of our representation results are found in [BGP07]. [Ben07c] contains some further DEL-style game analysis in its section on extensive games. Further studies in this line are [Har04] on preference and player's powers in games, [Bru04] on epistemic foundations of game theory, [Ott05] on update in games concerning players' strategic intentions and preferences, [DZ07] and [Dég07b]. Also [Roy08] on the role of intentions and information dynamics in games, which develops connections with the philosophy of action.

Also, the results in this chapter suggest a richer temporal perspective, where belief changes do not just concern partial past observations, but also expectations about the future. This calls for a merge of temporal logic, dynamic-epistemic logic, and belief revision. For epistemic logic proper, this has been done in [BP06], with important protocol-based extensions in [BGP07]. Merging temporal logic with belief revision is done in [Bon07], and see [Zve07] for further elaboration. Branching temporal versions of dynamic belief revision in DEL-style have been explored in [Dég07a]. Related work includes [BHT06], [HT06], etc.

Similarly, the logical style of analysis presented here needs to be brought into contact with the ways in which game theorists study bounded rationality (cf. [OR94], [Rub98]). These tie in more with complexity-theoretic diversity in processing capacities of players, and/or computational difficulty of the games they are playing (cf. [Sev06]).

Finally, we think that interaction of diverse agents is a topic with many logical repercussions, of which we have merely scratched the surface. But this is a topic which will calls for a yet more general perspective on sources of diversity, to be presented in the next chapter.