Fixed-point logics on trees
Gheerbrant, A.P.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
6.1 Game Solution as Rational Procedure

In this chapter, we will focus on finite games, which can be represented in so-called extensive form as special enriched tree structures. Logic and games form a natural combination. On the one hand, there are “logic games” that analyze basic notions such as truth, proof, or model comparison, while on the other hand, standard logical systems have proved applicable to many basic issues in the foundations of game theory (cf. [14], [114]). This chapter will concentrate on the second aspect.

Logics that describe games In recent years, many logical analyses have been given of both strategic and extensive games, through introducing formal languages that describe game structure while raising logical questions of definability and axiomatization ([29], [13], [51], [87]). A benchmark for logics in this tradition has been the definition of Backward Induction (“BI” for short), the most common method for solving finite extensive games of perfect information ([122], [123]). In this same arena, basic foundational results have been obtained in epistemic game theory, endowing bare games with epistemic assumptions about players. A pilot result was the characterization of the BI outcome in terms of assuming common knowledge, or common true belief, in rationality, meaning that players choose those actions that they believe to be best for themselves ([6]).

Analyzing solution procedures Recently, [18] has suggested that the main focus here should be shifted: away from a static assumption of known or believed rationality to the underlying “procedural rationality” of plausible procedures that players engage in when analyzing and playing a game, and the way these result
in stable limit models where rationality becomes common knowledge.\textsuperscript{1} Thus, [13] shows how game-theoretic equilibrium fits with the computational perspective of fixed-point logics, and [20] gives several dynamic procedures that analyze \( \text{BI} \). This chapter will analyze these proposals further, and find their common mathematical background. This will then be our starting point for suggesting a more general line of investigation.

**Basics of extensive games**  We assume some basic game theory, and we will work with \textit{finite extensive games} of perfect information, i.e., finite trees with labelled nodes, where each node is either an end node, or an intermediate node that represents the turn of a unique player.\textsuperscript{2} We will mostly think of 2-player games, though much of what we say generalizes to more players. While game trees with moves are simple computational structures, the essence of rational action arises with the way players evaluate outcomes. Thus, there is also a further \textit{preference relation} for each player between end nodes (encoding complete histories) that we will take to be a total order in this chapter, though this requirement could be generalized. Equivalently, such total evaluation orders may be represented in the form of numerical utility values for players at end nodes.

**Backward induction**  We now define our basic procedure in a bit more detail:

**Definition 6.1.1** (\( \text{BI} \) procedure for “generic” extensive games). We call a game \textit{generic} when, for each player, distinct end nodes have different utility values. On such games \textit{Backward Induction} is this inductive algorithm:

“At end nodes, players already have their values marked. At further nodes, once all daughters are marked, the player to move gets her maximal value that occurs on a daughter, while the other, non-active player gets his value on that maximal node.”

A \textit{strategy} for a player is a map that selects one move at each turn for that player. It is easy to see that \( \text{BI} \) generates a strategy for each player at her turns: go to the successor node that has your highest value. The resulting set of strategies is the \textit{“BI outcome”}, that leads to a unique play of the game. We will call the set-theoretic union of all these strategies (still a function on nodes) \( \text{bi} \). The \( \text{BI} \) procedure seems obvious, telling us players best course of action. And yet, it is packed with assumptions about how players behave that are worth highlighting. For now, just note that the algorithm subtly changes its interpretation of values on the way. At leaves, these values encode plain utilities or preferences, but at nodes higher up in the game tree, the \( \text{BI} \) values clearly mix in additional considerations of plausibility, incorporating beliefs about what others will do.

\textsuperscript{1}Note that even the common word “solution” has an ambiguity between a procedure (“Solution is not easy”) and a static product of such a procedure (“Show me your solution”).

\textsuperscript{2}Only towards the end, we will briefly consider games with imperfect information.
6.2. From Functional to Relational Strategies

Delicate cases  BI can produce debatable outcomes, as in the next illustration:

Example 6.1.2 (A simple BI outcome). In the following game, players’ preferences are encoded in utility values, as pairs (value for A, value for E). Backward Induction tells player E to turn left at her turn, which gives A a belief that this will happen, and so, based on this belief about his counter-player, A should turn left at the start, making both worse off than they might have been:3

The fact that the BI prediction or recommendation is not always intuitive has motivated much logical analysis of the procedure and the reasoning underpinning it. We will not enter this debate here. We neither endorse nor reject Backward Induction, but we merely take it as our point of entry into the logic of game solution procedures. Our starting points are three different proposals for explaining what makes BI tick, that we will explain in due course. But before getting there, let us first make a generalization of what we mean by Backward Induction.

6.2  From Functional to Relational Strategies

Strategies as subrelations of the move relation  A game-theoretic strategy is usually taken to be a function on nodes in a game tree, yielding a unique recommendation for play there. But in many settings, it makes more sense to think of strategies as nondeterministic binary subrelations of the total relation move (the union of all labelled actions in the game) that merely constrain further moves by selecting one or more as admissible. This is in line with the colloquial use of the term “strategy”, it also reflects a common view of plans for action, and technically, it facilitates logical definitions of strategies in propositional dynamic logic [19].

4People defend this outcome by saying that the game is “competitive”, but that amounts to giving information about the players that is not explicit in the game tree. If such extra information is relevant to solution, we may need a richer notion of game from the start.

4Frankly, we have dramatized things a bit here to catch the reader’s attention. Since the numbers just encode ordinal preferences, the same point might have been made with values 0, 1, 2 and 3. But the undesirable point remains that the computed outcome is not Pareto-optimal. An outcome of a game is Pareto optimal if there is no other outcome that makes every player at least as well off and at least one player strictly better off.
**Relational BI, first version** Indeed, one common numerical formulation of BI already has this relational flavor. We now (as for the remainder of the chapter) drop the assumption that games are generic:

**Definition 6.2.1** (Relational Backward Induction, first version). Starting from the leaves, one now assigns values for players at nodes using the rule:

Suppose that $E$ is to move at a node, and all values for daughters are known. The $E$-value is the maximum of all the $E$-values on the daughters, while the $A$-value is the minimum of the $A$-values at all $E$-best daughters.$^5$

The relation $bi$ arising from this algorithm connects nodes to all daughters with maximal values for the active player, of which there may be more than one. This method focuses on minimal values that can be guaranteed when doing the best within one’s power.

**Solution algorithms make assumptions about players** But while this looks like an obvious numerical rule, it does embody special assumptions about players. In particular, taking the minimum value is a worst-case assumption that my counter-player does not care about my interests after her own are satisfied. But we might also assume that she does, choosing among her maximum nodes one that is best for me. In that case, the second numerical value in the algorithm would be a maximum rather than a minimum. And other options are possible.$^6$

This variety of relational versions of game solution is not a problem. It rather highlights an important feature of game theory: mathematical “solution methods” are not neutral, they encode significant assumptions about players. But the variety does suggest that we start by finding a general base version of BI that is not too specific:

**A minimal notion of rationality: avoid stupid moves** Here is one logical analysis of the variety for relational versions of BI. Let us first view matters from a somewhat higher standpoint. Suppose that I need to compare different moves of mine, each of which, given the relational nature of the procedure, still allows for many leaves (end nodes) that can be reached via further $bi$-play.$^7$ A minimal notion of Rationality would then say that

*I do not play a move when I have another move whose outcomes I prefer.*

---

$^5$The dual calculation for values at A’s turns is completely analogous.

$^6$Of course, one might view such alternatives as calling for a change in players’ utilities. We will not get into this perennial issue of game preference transformations here.

$^7$In this perspective with total outcomes of the game, we make a shift from the original version of the BI algorithm, which looked at daughters of the current node only.
6.2. From Functional to Relational Strategies

A source of variety: different set preferences  This seems plausible, but what notion of preference is involved here? It is easy to see that, in the above first version of the BI algorithm, the following choice is made. Player $i$ preferred a set $Y$ of leaves reachable by further bi-play to another set $X$ if the minimum of its values for $i$ is higher. This means that we have the following $\forall \exists$ pattern for set preference:

$$\forall y \in Y \exists x \in X : x <_{i} y$$

But clearly, staying with the same over-all notion of Rationality, there are several alternatives for comparisons between reachable sets of outcomes. One common notion of preference for $Y$ over $X$ in the logical literature ([135], [105]) is the $\forall \forall$ stipulation that

$$\forall y \in Y \forall x \in X : x <_{i} y$$

Relational backward induction, second version  Clearly, avoiding moves that should not be taken under this stronger notion of preference is a weaker constraint on behavior of players. Still, it fits with a minimal game-theoretic solution procedure for strategic games called eliminating strictly dominated strategies ([112]). We will take this second relational version of Backward Induction as our running example:

Definition 6.2.2 (Relational Backward Induction, second version). First, mark all moves as “active”. Call a move to a node $x$ dominated if $x$ has a sibling from which all reachable endpoints via active moves are preferred by the current player to all reachable endpoints via active moves from $x$ itself. The second version of the BI algorithm works in stages:

At each stage, it marks dominated moves in the $\forall \forall$ sense of set preference as “passive”, leaving all others active. In this preference comparison between sets of outcomes, the “reachable endpoints” by an active move are all those that can be reached via a sequence of moves that are still active at this stage.

In another well-known terminology, players play a “best response”.

Henceforth, we will use BI to refer to this algorithm, and the subrelation of the total move relation produced by it at the end. It is a cautious notion of game solution making fewer assumptions about the behavior of other agents than the earlier version. Of course, the two versions agree on generic games, for which the subset of the move relation obtained as output is always a function.

---

Given that we have finite total orders, we could also replace this by

$$\exists x \in X \forall y \in Y : x <_{i} y$$
Example 6.2.3 (Some comparisons). Consider the following two games, where the values indicated are utilities for player $A$. For simplicity, we assume that player $E$ has no preference between her moves:

In the game to the left, our first version of Backward Induction makes $A$ go right, since the minimum 2 is greater than the 1 on the left. But our cautious BI will accept both moves for $A$, as no move strictly dominates the other.

Moreover, both versions will accept all moves in the game to the right. This may seem strange, since most players would probably go right at the start: they have nothing to lose, and a lot to gain. But analyzing all variants for preference comparisons between sets of outcomes is not our focus here. We will return to the issue of further possible solution concepts in later sections.

Important remark. Our style of analysis chooses one particular line toward generalizing Backward Induction to non-generic games. But others make sense, too, as pointed out by Cédric Dégremont. For instance, if one thinks of strategy profiles in Nash equilibrium, the following game would have two:

Both profiles $(a, d)$ and $(b, c)$ are in equilibrium. But our algorithm will leave both options for $E$, and tell $A$ to go left. This chapter will not address the alternative analysis of the BI-output in terms of sets of strategy profiles, leaving this as a challenge to fixed-point logics over richer models.

6.3 Defining BI as a Unique Static Relation

Many definitions for the BI relation on generic games have been published by logicians and game-theorists (cf. the survey in [55]). Our point of departure here is a version involving a modal language of $a$-labelled moves, i.e., binary transition relations $a$ on nodes with matching modalities $\langle a \rangle$, plus a modal preference operator interpreted as follows at nodes of a game tree:
6.3. Defining BI as a Unique Static Relation

\[ \langle \text{pref}_i \rangle \varphi: \text{player } i \text{ prefers some node where } \varphi \text{ holds to the current one} \]

The original result Here is a result from [23]:

**Theorem 6.3.1.** On generic games, the BI strategy is the unique function \( \sigma \) which is total on non-terminal nodes and satisfies the following modal axiom for all propositions \( p \) - viewed as sets of nodes - for all players \( i \):

\[
(turn_i \land \langle \sigma^* \rangle (end \land p)) \rightarrow [\text{move}] \langle \sigma^* \rangle (end \land \langle \text{pref}_i \rangle p)
\]

For a proof (a laborious but straightforward induction on finite tree depth), we refer to the cited paper. Here we just concentrate on the meaning of the crucial axiom, that may be brought out by a standard modal frame correspondence, where frame truth quantifies universally over all sets of objects for proposition letters ([26]). The frames here are games extended with one more binary relation \( \sigma \). What we find is a notion of **Rationality** like before:

**Fact 6.3.2.** An extended game makes \((turn_i \land \langle \sigma^* \rangle (end \land p)) \rightarrow [\text{move}] \langle \sigma^* \rangle (end \land \langle \text{pref}_i \rangle p)\) true for all \( i \) at all nodes iff it has this property for all \( i \):

**RAT-1:** No other available move for the current player \( i \) yields a set of outcomes by further play using \( \sigma \) that has a higher minimal value for \( i \) than the outcomes of playing \( \sigma \) all the way down the tree from the current node.

**Proof.** This is a standard modal correspondence argument that we omit. The correspondence language uses the reflexive-transitive closure of the relation \( \sigma \), but this is a simple extension of known techniques ([16]). \( \square \)

The typical picture to keep in mind here, and also later on, is this:

\[
\begin{array}{c}
\sigma \\
\text{via } \sigma \\
\text{via } \sigma \\
\text{via } \sigma \\
\geq \\
\end{array}
\]

RAT-1 is equivalent to this *confluence property* for action and preference:
Chapter 6. Fixed-point Logics on Finite Extensive Games

CF1 : \( \bigwedge_i \forall x (\text{turn}_i(x) \rightarrow \forall y (\sigma(x, y) \rightarrow (\text{move}(x, y) \land \forall u ((\text{end}(u) \land \sigma^*(y, u)) \rightarrow \forall z (\text{move}(x, z) \rightarrow \exists v (\text{end}(v) \land \sigma^*(z, v) \land v \leq_i u)))))) \)

This \( \forall \forall \exists \) form is a comparison between sets of outcomes that negates an earlier notion of preference: the minimum value on the reachable endpoints after \( z \) is not larger than that after \( y \). It is easy to show that any relation \( \sigma \) which assigns a successor to each non terminal node and which satisfies this property matches the BI solution level by level on generic games.

Capturing BI in logical terms But now let us look at our favored relational generalization of BI. First, we reformulate the stated non-dominance property:

RAT-2. No alternative move for the current player \( i \) guarantees outcomes via further play using \( \sigma \) that are all strictly better for \( i \) than all outcomes resulting from starting at the current move and then playing \( \sigma \) all the way down the tree.

A logical formula defining this has the following \( \forall \forall \exists \) form:

CF2 : \( \bigwedge_i \forall x \forall y ((\text{turn}_i(x) \land \sigma(x, y)) \rightarrow (\text{move}(x, y) \land \forall z (\text{move}(x, z) \rightarrow \exists u \exists v (\text{end}(u) \land \text{end}(v) \land \sigma^*(y, v) \land \sigma^*(z, u) \land u \leq_i v)))) \)

Theorem 6.3.3. BI is the largest subrelation of the move relation in a finite game tree satisfying the two properties that (a) the relation has a successor at each non-terminal node, and (b) CF2 holds.10

Proof. First, the given algorithm clearly leaves at least one active move at each node, by the definition of preference. Moreover, at the final state, when no more deactivations occur, CF2 must hold: there are no more dominated moves, and that is what it says.

That the relation defined in this way is maximal may be seen as follows. If we reactivate anywhere a move that is inactive, that move had disappeared at some stage because it was dominated there by another move. But then it would still be dominated in the whole tree by the same move. For, all that can have happened in the further stages of the algorithm is that fewer endpoints have become reachable through active paths from the two moves, and their \( \forall \forall \) -dominance relationship then persists.

Conversely, if we have any subrelation of the move relation with the given two properties, it is easy to see by induction on the depth of subtrees that all its moves survive each stage of the above main BI procedure, by the definition of the elimination step.

9One could change the formal language for CF1 here to a more technical first-order one avoiding the closure operator – but for our main points, such variations are not important.

10We say “largest” in this formulation because in the presence of more than one best successor, different subrelations of the move relation might satisfy CF2. Note that there need not be a largest relation satisfying a given structural property, but in this particular case, it does.
6.4. A Dynamic-Epistemic Scenario: Iterated Announcement of Rationality

We now make these same points about the procedure more syntactically, by inspecting the syntax of \( \text{CF2} \). We can restate this in terms of the well-known formalism of first-order least fixed-point logic \( \text{FO(LFP)} \):\(^{11}\)

**Theorem 6.3.4.** The BI relation is definable in \( \text{FO(LFP)} \).

**Proof.** Indeed, the definition involves just one *greatest fixed-point* in addition to the transitive closure operations. This fixed-point is in the language of \( \text{FO(LFP)} \), all occurrences of the predicate symbol \( X \) in the relevant formula are positive:

\[
\text{BI}(x, y) = [GFP_{X, x, y}(\text{move}(x, y) \land \bigwedge_i (\text{turn}_i(x) \rightarrow \forall z (\text{move}(x, z) \\
\rightarrow \exists u \exists v (\text{end}(u) \land \text{end}(v) \land X^*(y, v) \land X^*(z, u) \land u \leq v))))(x, y)]^{12}
\]

(Note that we use \( X^*(z, u) \) as shorthand for \( [TC_{z, u}X(z, u)](z, u) \), which is expressible in \( \text{FO(LFP)} \), see Chapter 2.) \( \square \)

This definition will be our point of reference in what follows. Interestingly, it is both a static description of the BI relation and also a definition of a procedure computing it. For, we can now use the standard defining sequence for a greatest fixed-point, starting from the total move relation, and see that its successive decreasing approximation stages \( X^k \) are exactly the “active move stages” of the above algorithm. We will refer to these stages \( X^k \) at several places in what follows. In our view, fixed-point logics are attractive since they analyze both the statics and dynamics of game solution.

In the following sections, we extend this theme by looking at two further logical ways of construing the Backward Induction procedure that have been proposed in recent years.

### 6.4 A Dynamic-Epistemic Scenario: Iterated Announcement of Rationality

Here is another procedural line on Backward Induction as a rational process. \cite{18} proposed an analysis in the spirit of current *dynamic-epistemic logics* that describe acts of information flow, such as public announcements or observations \((\cite{56}, \cite{21})\). The following analysis of BI takes it to be a process of prior off-line deliberation about a game by players whose minds proceed in harmony - though they need not communicate in reality.\(^{13}\)

\(^{11}\)In terms of \cite{15}, the syntax of \( \text{CF2} \) has dual “PIA form”, guaranteeing that the union of all relations satisfying \( \text{CF2} \) exists, while a small extra argument gives the existence.

\(^{12}\)We can also replace the reflexive transitive closures \( X^* \) by definitions in \( \text{FO(LFP)} \).

\(^{13}\)Compare also the dynamic agreement procedures studied in \cite{68}.
Solving games by announcements of rationality  The following analysis uses the dynamic epistemic logic of public announcements of the form $!\varphi$, which say that some proposition $\varphi$ is true. These transform a current epistemic model $\mathcal{M}$ into its submodel $\mathcal{M}[\varphi]$ whose domain consists of just those worlds in $\mathcal{M}$ that satisfy $\varphi$. [18] makes the solution process of extensive games itself the focus of a PAL style analysis:

**Definition 6.4.1** (Node rationality). As before, at a turn for player $i$, a move to a node $x$ is *dominated* by a move to a sibling $y$ of $x$ if every history through $x$ ends worse, in terms of $i$’s preference, than every history through $y$. Now $\text{rat}$ says that “at the current node, no player has chosen a strictly dominated move in the past coming here”.

This makes an assertion about nodes in a game tree, viz. that they did not arise through playing a dominated move. Some nodes will satisfy this, others may not. Note that we do not say that every node in the game satisfies $\text{rat}$: we merely say that it is an informative property of nodes. Thus, announcing this formula as a fact about the players of a game is informative, and it will in general make the current game tree smaller.

But then we get a dynamics as in famous puzzles like the Muddy Children, where *repeated assertions* of ignorance eventually produce enough information to solve the whole puzzle. In our case, in the new smaller game tree, new nodes may become dominated, and hence announcing $\text{rat}$ again (saying that it *still* holds after this round of deliberation) makes sense, and so on. This process of iterated announcement must always reach a limit, that is, a smallest subgame where no node is dominated any more:

**Example 6.4.2** (Solving games through iterated assertions of Rationality). Consider a game with three turns, four branches, and pay-offs for $A$, $E$ in that order:

![Game Tree]

Stage 0 of the procedure rules out point $u$ (the only point where Rationality fails), Stage 1 rules out $z$ and the node above it (the new points where Rationality fails), and Stage 2 rules out $y$ and the node above it. In the remaining game, Rationality holds throughout:
6.4. A Dynamic-Epistemic Scenario: Iterated Announcement of Rationality

In such generic games, the BI solution emerges step by step. [18] shows that the actual Backward Induction path for extensive games is obtained by repeated announcement of the assertion rat to its limit. We repeat some relevant notions from dynamic-epistemic logic:

Definition 6.4.3 (Announcement limit). For each epistemic model \( M \) and each proposition \( \varphi \) that is true or false at points in the model, the announcement limit \((\varphi, M)^\#\) is the first model reached by successive announcements \( \varphi! \) that no longer changes after the last announcement is made.

That such a limit exists is clear for finite models, since the sequence of sub-models is weakly decreasing. There are two possibilities for the limit model. Either it is non-empty, in which case \( \varphi \) holds in all nodes, meaning that it has become common knowledge (the self-fulfilling case), or it is empty, meaning that the negation \( \neg \varphi \) has become common knowledge (the self-refuting case). Both occur in concrete puzzles, though generally speaking, rationality assertions like rat tend to be self-fulfilling, while the ignorance statement that drives the Muddy Children is self-refuting: at the end, it holds nowhere.

Capturing BI by iterated announcements With general relational strategies, the iterated announcement scenario produces the earlier \( \forall \exists \exists \) version of Backward Induction:

Theorem 6.4.4. In any game tree \( M \), \((\text{rat}, M)^\#\) is the actual subtree computed by BI.

This can be proved directly, but it also follows from our next observations. For a start, it turns out easier to change the definition of the driving assertion rat a bit. We now consider rat’, which only demands that the current node was not arrived at directly via a dominated move for one of the players. This does not eliminate nodes further down, and indeed, announcing this repeatedly will make the game tree fall apart into a forest of disjoint subtrees – as is easily seen in the above examples. These record more information.

\[\text{Announcement limits also exist in infinite models, if one takes intersection at limit ordinals.}\]
Sets of nodes as relations Here is an obvious fact about game trees. Each subrelation $R$ of the total move relation has an obvious unique corresponding set of nodes $\text{reach}(R)$ consisting of the set-theoretic range of $R$ plus the root of the tree (we add the latter for convenience). And vice versa, each set $X$ of nodes induces a unique corresponding subrelation of the move relation $\text{rel}(X)$ consisting of all moves in the tree that end in $X$. Incidentally this suggests that Theorem 6.3.4 can be slightly refined:

**Theorem 6.4.5.** The BI relation is definable in $\text{FO}(\text{LFP}^1)$.

**Proof.** We simply put

$$\text{Bl}(x, y) = \text{Move}(x, y) \land [\text{GFP}_{x,y} \exists x(\text{move}(x, y) \land \bigwedge_i (\text{turn}_i(x) \rightarrow \forall z(\text{move}(x, z)$$

$$\rightarrow \exists u \exists v ( [\text{TC}_{zu} \text{Move}(z,u) \land X(u)](z,u) \land [\text{TC}_{yv} \text{Move}(y,v) \land X(v)](y,v)$$

$$\land \text{end}(u) \land \text{end}(v) \land u \leq_i v)))](y)$$

With this simple connection, we can link the earlier approximation stages $\text{BI}^k$ for Backward Induction (i.e., the successive relations computed by our earlier procedure) and the stages of our public announcement procedure. They are in harmony all the way:

**Fact 6.4.6.** For each $k$, in each game model $\mathcal{M}$, $\text{BI}^k = \text{rel}((\text{\textsc{rat}}')^k, \mathcal{M})$.

**Proof.** By induction on $k$. The base case is obvious: $\mathcal{M}$ is still the whole tree, and the relation $\text{BI}^0$ equals move. Next, consider the inductive step. If we announce $\text{rat}'$ again, we remove all points reached by a move that is dominated for at least one player. These are precisely the moves cancelled by the corresponding step of the BI algorithm.

It follows also that, for each stage $k$, $\text{reach}(\text{BI}^k) = ((\text{\textsc{rat}}')^k, \mathcal{M})$.

Either way, we conclude that the earlier algorithmic fixed-point definition of the BI procedure and van Benthem’s iterated announcement procedure amount to the same thing.  

Thus, one might say that the deliberation scenario is just a way of “conversationalizing” the underlying mathematical fixed-point computation. Still, it is of interest in the following sense. Viewing a game tree as an epistemic model with nodes as worlds, we see how repeated announcement of Rationality eventually makes this property true throughout the remaining limit model: in this way, it has made itself into common knowledge.

---

15We leave the technical question open to which extent this is a more general technical method for switching between different types of predicate arities with fixed-points.
6.5 Another Dynamic Scenario: Beliefs and Iterated Plausibility Upgrade

Next, in addition to knowledge, consider the equally fundamental notion of belief. Many foundational studies in game theory (cf. the extensive discussion and references for belief-based game theory in [137]) view Rationality as choosing a best action given what one believes about the current and future behavior of the players. Indeed, this may be the most widely adopted view of game solution in the epistemic foundations of game theory today. We will first state a logical analysis of game solution in these terms, and then relate it to our earlier account of Backward Induction.

**Backward Induction in a soft light** An appealing take on the BI strategy in terms of beliefs uses “soft update” that does not eliminate worlds as above for announcements !$\varphi$, but rearranges the plausibility order between worlds.\(^{16}\) A typical example is the radical upgrade $\uparrow \varphi$ that makes all current $\varphi$-worlds best, and then puts all $\neg \varphi$-worlds underneath, while keeping the old ordering inside these two zones. Now recall our earlier observation that Backward Induction really creates expectations for players. All the essential information produced by the algorithm is then in the binary plausibility relations that it creates inductively for players among non terminal nodes in the game, standing for complete histories. To see this, consider our running example once more:

**Example 6.5.1** (The debatable BI outcome, hard and soft). The hard scenario in terms of events !rat removes nodes $x$ from the tree that are reached via moves which are strictly dominated by moves to siblings of $x$ as long as this can be done, resulting in the following sequence of stages:

By contrast, a soft scenario does not remove nodes but modifies the plausibility relation. We start with all endpoints of the game tree incomparable (other versions would have them equiplausible). Next, at each stage, we compare sibling nodes, using this notion:

\(^{16}\)This is a technique from current dynamic epistemic logics, where acts of knowledge update and belief revision are represented by transformations on domains of models, or their plausibility ordering of worlds. See [17], [9], [8] for some latest developments.
Definition 6.5.2 (Rationality in beliefs). A move to a node $x$ for player $i$ dominates a move to a sibling $y$ of $x$ in beliefs if the most plausible end nodes reachable after $x$ along any path in the whole game tree are all better for the active player than all the most plausible end nodes reachable in the game after $y$. Rationality* ($\text{rat}^*$) is the assertion that no player plays a move that is dominated in beliefs.

Now we perform a relation change that is like a radical upgrade $\uparrow \text{rat}^*$, except that plausibility upgrades may take place in subtrees, and hence one needs to work with submodels of the whole set of histories:

If a move to a node $x$ dominates a move to a sibling $y$ of $x$ in beliefs, we make all end nodes reachable from $x$ more plausible than those reachable from $y$, keeping the old order inside these zones.

This changes the plausibility order, and hence the dominance pattern, so that belief statements can change their truth values – and a genuine iteration can start. Here are the stages for this procedure in the above example, where we use the letters $x, y, z$ to stand for the end nodes or histories of the game:

\[
\begin{array}{c}
\text{A} & 1,0 & \text{E} & 0,100 & 99,99 \\
\text{x} & y & z
\end{array}
\]

In the first game tree, going right is not yet dominated in beliefs for $A$ by going left. $\text{rat}^*$ only has bite at $E$'s turn, and an upgrade takes place that makes $(0,100)$ more plausible than $(99,99)$. After this upgrade, however, going right has now become dominated in beliefs, and a new upgrade takes place, making $A$’s going left most plausible.\footnote{Here the plausibility relation is defined over end nodes only. Another option would have been to define it over all nodes in the tree (see [13]).}

Theorem 6.5.3. On finite trees, the Backward Induction strategy is encoded in the final plausibility order for end nodes created by iterated radical upgrade with rationality in belief.

At the end of this procedure, players have acquired common belief in rationality. Let us now prove the result, using an idea from [10].
Strategies as Special Plausibility Relations  We present now in details a set-theoretical transformation which allows to go back and forth between strategies and plausibility relations in the case of finite extensive games. To do so, we characterize plausibility orders as a special sort of linear order over leaves, which satisfy a property that we call “tree compatibility”.

Let \( a \) be a node in a finite tree. A history containing \( a \) is a path along the Move relation which contains \( a \), starts from the root and ends at a leaf. We denote by \( RL(a) \) the set of leaves reachable via a history containing \( a \). Note that every leaf in the tree determines a unique history (given a leaf, there is a unique path starting from the root and ending at this leaf).

**Definition 6.5.4 (Relational strategy).** A relation \( \text{Best} \) on a finite game tree \( \mathcal{M} \) is a relational strategy whenever:

- \( \text{Best} \subseteq \text{Move} \)
- every non-terminal node is in the \( \text{Best} \) relation with another node in the tree, i.e., \( \mathcal{M} \models \forall x (\neg \text{end}(x) \rightarrow \exists y \text{Best}(x,y)) \)

Some linear orders over leaves satisfy some special conditions which ground their “equivalence” (in a sense to be shown below) with relational strategies. Such linear orders satisfy a property that we call *tree-compatibility*.

**Definition 6.5.5 (Ancestor-connected sets of leaves).** Let \( A, B \) be two sets of leaves in a finite tree. \( A \) and \( B \) are ancestor-connected if there exists a node \( x \) with two children \( z \) and \( y \) such that the set of \( z \)-reachable leaves is exactly \( A \) and the set of \( y \)-reachable leaves is exactly \( B \).

**Definition 6.5.6 (Tree-compatible linear order).** Let \( \preceq \) be a linear order on the leaves of a finite tree. \( \preceq \) is tree-compatible if for all ancestor-connected sets \( A, B \) in the tree, either every leaf in \( A \) is below every leaf in \( B \) relative to \( \preceq \), or every leaf in \( B \) is below every leaf in \( A \) relative to \( \preceq \).

As an example, there can be no criss-crossing as in the following tree:

```
     ●
   /   \
  ●    ●
 /  \   /
●  ●  ●
  x  z  y
    /  /
   /  u
   with x < y < z < u
```

A very natural example of tree-compatible linear order in a finite tree is the left-right order over its leaves.

The following simple lemma will be useful in the proof of Theorem 6.5.8. It states that if one consider the set of least common ancestors of every two pairs of leaves formed out of three leaves, it cannot contain more than two distinct nodes.
Lemma 6.5.7. Let $u$, $v$, $w$ be three leaves in a finite tree, $a$ the least common ancestor of $u$ and $v$ and $a'$ the least common ancestor of $v$ and $w$. Then, consider $a''$, the least common ancestor of $u$ and $w$. Then either $a = a'$ or $a' = a''$.

Proof. Consider the unique histories $U$, $V$ determined respectively by $u$ and $v$. The two paths start together at the root and there is a unique point where they diverge. This point is $a$, the least common ancestor of $u$ and $v$. Then, consider the unique history $W$ determined by $w$. Similarly, $W$ crosses $V$ on the unique point where they start to diverge, which is $a'$. Now there are two cases, either $U$ and $W$ start to diverge after $U$ and $V$ do, or they start before. In the first case $a' = a$ and in the second case, $a' = a''$. \hfill $\Box$

We first show that each relational strategy $BI$ induces a tree-compatible linear order $\preceq_{\text{Best}}$ on leaves as follows: $x \preceq_{\text{Best}} y$ iff whenever $y$ is reached via a $\text{Best}$ move from the least common-ancestor of $x$ and $y$, then so is $x$.

Theorem 6.5.8 (From strategies to plausibility orders). Let $\text{Best}$ be a relational strategy and let $\preceq_{\text{Best}}$ be the smallest binary relation on the leaves such that for every $(a, b) \in \text{Move}$, $(a, c) \in \text{Move}$, $u \in \text{RL}(b)$ and $v \in \text{RL}(c)$:

- $(a, c) \notin \text{Best}$ implies $u \preceq_{\text{Best}} v$
- $(a, c) \in \text{Best}$ implies $v \preceq_{\text{Best}} u$

The defined relation $\preceq_{\text{Best}}$ is a tree-compatible linear order and we say that it corresponds to $\text{Best}$.

Proof. Consider a relational strategy $\text{Best}$ on a finite tree together with the corresponding relation $\preceq_{\text{Best}}$.

First assume that $\preceq_{\text{Best}}$ is not total. Then there are two leaves $u$ and $v$ which are not connected in $\preceq_{\text{Best}}$. Let $a$ be their least common ancestor, $b$ its immediate successor on the path to $u$ and $c$ its immediate successor on the path to $v$. Consider $(a, b)$ and $(a, c)$. There are four possible patterns as regards their inclusion in $\text{Best}$. In each case, by the definition of $\preceq_{\text{Best}}$, $u$ and $v$ stand together in the $\preceq_{\text{Best}}$ relation.

Now assume $\preceq_{\text{Best}}$ is not transitive. Then there are three leaves $u$, $v$, $w$ such that $u \preceq_{\text{Best}} v$, $v \preceq_{\text{Best}} w$ and $w \preceq_{\text{Best}} u$. Let $a$ be the least common ancestor of $u$ and $v$, $b$ its immediate successor on the path to $u$ and $c$ its immediate successor on the path to $v$. Let $a'$ be the least common ancestor of $v$ and $w$, $b'$ its immediate successor on the path to $v$ and $c'$ its immediate successor on the path to $w$. Let $a''$ be the least common ancestor of $u$ and $w$, $b''$ its immediate successor on the path to $u$ and $c''$ its immediate successor on the path to $w$. By definition of $\preceq_{\text{Best}}$, the following holds:

- $(a, c) \notin \text{Best}$ or $(a, b) \in \text{Best}$
6.5. Another Dynamic Scenario: Beliefs and Iterated Plausibility Upgrade

• \((a', b') \in \text{Best} \) or \((a', c') \notin \text{Best} \)

• \((a'', c'') \in \text{Best} \) and \((a'', b'') \notin \text{Best} \)

Now we want to consider the least common ancestor of \(u, v\) and \(w\). By Lemma 6.5.7, there are three cases. It could either be \(a, a'\) or \(a''\). Without loss of generality (the reasoning is similar in the other two cases), let us assume it is \(a''\). Then, again by Lemma 6.5.7, there are two cases, either \(a'' = a\) (and \(b'' = b\), \(c'' = c\)), i.e., it is also the least common ancestor of \(u, v\), either \(a'' = a'\) (and \(b'' = b', c'' = c'\)), i.e., it is also the least common ancestor of \(v, w\). Let us assume the latter (again, the former case is similar). It follows that:

• \((a'', b'') \in \text{Best} \) or \((a'', c'') \notin \text{Best} \)

• \((a', c') \in \text{Best} \) and \((a', b') \notin \text{Best} \)

But this contradicts our assumptions.

Finally, assume \(\preceq_{\text{Best}}\) is not tree-compatible. So we can assume without loss of generality that there are two ancestor-connected sets of leaves \(A\) and \(B\), with \(u_1, u_2 \in A\) and \(v_1, v_2 \in B\) such that \(u_1 \preceq_{\text{Best}} v_1\) and \(v_2 \preceq_{\text{Best}} u_2\). Let \(a\) be the least common ancestor of the leaves in \(A\) and \(B\), \(b\) its immediate successor on the paths to the leaves in \(A\) and \(c\) its immediate successor on the paths to the leaves in \(B\). As \(u_1 \preceq_{\text{Best}} v_1\), by construction of \(\preceq_{\text{Best}}\), \((a, b) \in \text{Best}\) and \((a, c) \notin \text{Best}\). Similarly, by construction of \(\preceq_{\text{Best}}\), as \(v_2 \preceq_{\text{Best}} u_2\), it follows that \((a, b) \notin \text{Best}\), which is a contradiction.

Now, conversely, any tree-compatible linear order \(\preceq\) on leaves induces a relational strategy \(\text{Best}_{\preceq}\) defined by selecting just those available moves at a node \(z\) that have the following property: their further available histories lead only to \(\preceq\)-minimal leaves in the total set of leaves that are reachable from \(z\).

**Theorem 6.5.9** (From plausibility orders to strategies). Let \(\preceq\) be a tree-compatible linear order on a finite tree and let \(\text{Best}_{\preceq}\) be the smallest binary relation on the tree which satisfies the following: for every \((a, b) \in \text{Move}\), \((a, b) \in \text{Best}_{\preceq}\) whenever there is a leaf \(u\) reachable from \(b\) such that for every leaf \(v\) reachable from \(a\), \(u \preceq v\). Then \(\text{Best}_{\preceq}\) is a relational strategy and we say that it corresponds to \(\preceq\).

**Proof.** Consider a tree-compatible linear order over leaves \(\preceq\) on a finite tree together with the corresponding relational strategy \(\text{Best}_{\preceq}\). It is immediate that \(\text{Best}_{\preceq} \subseteq \text{Move}\). Now suppose \(\text{Best}_{\preceq}\) is not a relational strategy, i.e., there is a non-terminal node \(a\) which does not have any \(\text{Best}_{\preceq}\)-successor. By linearity of \(\preceq\), there is a leaf \(u\) reachable from \(a\) such that for every leaf \(v\) reachable from \(a\), \(u \preceq v\). Consider now the immediate successor \(b\) of \(a\) on the history generated by \(u\). By construction of \(\text{Best}_{\preceq}\), \((a, b) \in \text{Best}_{\preceq}\). \(\square\)

**Theorem 6.5.10.** Let \(\preceq\) be a tree-compatible linear order and \(\text{Best}_{\preceq}\) the corresponding relational strategy. Then \(\preceq = \preceq_{\text{Best}_{\preceq}}\).
Proof. Let \((u, v)\) be the least common ancestor of \(u\) and \(v\) and \(a\) the immediate successor of \(u\) on the history determined by \(u\). By construction of \(Best_{\leq}\), \((a, b)\) \(\in\) Best_{\leq} and by construction of \(\leq_{\text{Best}_{\leq}}\), \((u, v)\) \(\in\) \(\leq_{\text{Best}_{\leq}}\).

Let \((u, v)\) \(\in\) \(\leq_{\text{Best}_{\leq}}\). Similarly, let \(a\) be the least common ancestor of \(u\) and \(v\) and \(b\) the immediate successor of \(a\) on the history determined by \(u\). By construction of \(\text{Best}_{\leq}\), \((a, b)\) \(\in\) Best_{\leq} and \((u, v)\) \(\in\) \(\leq\).

Theorem 6.5.11. Let \(S\) be a relational strategy and \(\leq_{\text{S}}\) the corresponding tree-compatible linear order. Then \(S = \text{Best}_{\leq_{\text{S}}}\).

Proof. Let \((a, b)\) \(\in\) \(S\). Then, by construction of \(\leq_{\text{S}}\), for every \((a, c)\) \(\in\) Move, \(u \in RL(b)\) and \(v \in RL(c)\), \((a, c) \notin S\) implies \(u \leq_{\text{S}} v\) and \((a, c) \in S\) implies \(v \leq_{\text{S}} u\). By construction of \(\text{Best}_{\leq_{\text{S}}}\), it follows that \((a, b)\) \(\in\) \(\text{Best}_{\leq_{\text{S}}}\).

Let \((a, b)\) \(\in\) \(\text{Best}_{\leq_{\text{S}}}\). Then, by construction of \(\leq_{\text{S}}\), for every \((a, c)\) \(\in\) Move, \(u \in RL(b)\) and \(v \in RL(c)\), \((a, c) \notin S\) implies \(u \leq_{\text{S}} v\) and \((a, c) \in S\) implies \(v \leq_{\text{S}} u\). By construction of \(\text{Best}_{\leq_{\text{S}}}\), it follows that \((a, b)\) \(\in\) \(S\).

Via Theorems 6.5.11 and 6.5.10, the following definition gives a precise meaning to the assertion in [10] that “strategies are the same as plausibility relations”.

Definition 6.5.12. We say that a tree-compatible linear order \(\leq\) and a relational strategy \(\text{Best}\) are equivalent whenever \(\leq_{\text{S}} = \leq_{\text{Best}_{\leq}}\), or, equivalently, \(\text{Best}_{\leq} \leq_{\text{S}} \text{Best}_{\leq}\).

Now we can relate the computation in our upgrade scenario for belief and plausibility to the earlier relational algorithm for B1. Things are in harmony stage by stage:

Fact 6.5.13. For any game tree \(\mathcal{M}\) and any \(k\), \(((\uparrow \text{rat}^*)^k, \mathcal{M}))_{\text{Best}} = \text{B}1^k\).

Proof. The key point is as demonstrated in the earlier example of a stepwise B1 solution procedure. When computing a next approximation for the B1-relation according to CF2, we drop those moves that are dominated by another available one. But this has the same effect as making the leaves reachable from dominated moves less plausible than those reachable from surviving moves. And that was precisely the earlier upgrade step.

This structural equivalence also yields immediate matching syntactic transformations as follows.

Proposition 6.5.14. Let \(\text{Best}\) be a relational strategy, the corresponding plausibility relation \(\leq_{\text{Best}}\) can be defined as follows:

\[ y \leq_{\text{Best}} z \]
Another Dynamic Scenario: Beliefs and Iterated Plausibility Upgrade

\[ \exists x (\text{LCA}(x, y, z) \land (\exists w (\text{Best}(x, w) \land \text{move}^*(w, z)) \rightarrow \exists w' (\text{Best}(x, w') \land \text{move}^*(w', y)))) \]

where \( \text{LCA}(x, y, z) \) stands for “\( x \) is the least common ancestor of the leaves \( y \) and \( z \)”:\

\[ \text{LCA}(x, y, z) = \text{move}^*(x, y) \land \text{move}^*(x, z) \land \text{end}(y) \land \text{end}(z) \land \neg \exists w (w \neq x \land \text{move}^*(x, w) \land \text{move}^*(w, y) \land \text{move}^*(w, z)) \]

Proof. This follows by analyzing the proof of Theorem 6.5.8.

**Proposition 6.5.15.** Let \( \preceq \) be a tree-compatible linear order, the corresponding relational strategy \( \text{Best}_{\preceq} \) can be defined as follows:

\[ \text{Best}_{\preceq}(x, y) = \text{move}(x, y) \land \exists z (\text{move}^*(y, z) \land \text{end}(z) \land \forall z' ((\text{end}(z') \land \text{move}^*(x, z')) \rightarrow z \preceq z')) \]

Proof. This follows by analyzing the proof of Theorem 6.5.9.

Note that whenever a relational strategy \( \text{Best} \), or a plausibility order \( \preceq \) are definable in a logic \( L \) extending FO(TC^1), then it follows from the syntactic translations given in Proposition 6.5.14 and 6.5.15 that \( \preceq_{\text{Best}} \) and \( \text{Best}_{\preceq} \) are respectively also definable in \( L \).

**Theorem 6.5.16.** The binary relation \( \preceq_{\text{BI}} \) is definable in FO(LFP^1).

Proof. As BI is definable in FO(LFP^1), it follows from Proposition 6.5.14 that:

\[ y \preceq_{\text{BI}} z \]

\[ = \]

\[ \exists x (\text{LCA}(x, y, z) \land (\exists w (\text{BI}(x, w) \land \text{move}^*(w, z)) \rightarrow \exists w' (\text{BI}(x, w') \land \text{move}^*(w', y)))) \]

Proof. This follows by analyzing the proof of Theorem 6.5.8.

**Remark 6.5.17.** Let us point out that Theorem 6.5.16 could also be shown by providing a more direct definition in FO(LFP) as follows. For the sake of readability, let us first introduce the following shorthand (\( RL(u, x) \) stands for \( x \) is a leaf which is reachable from the node shorthand):\

\[ RL(u, x) : \approx \text{Move}^*(u, x) \land \text{end}(x) \]

Now we put:
Chapter 6. Fixed-point Logics on Finite Extensive Games

\[ x \preceq_{\text{BI}} y \]

= 

\[ [\text{GF}P_{x,y,X} \bigvee \exists z \exists u \exists v (\text{Turn}_i(z) \land \text{Move}(z,u) \land \text{Move}(z,v) \land \text{RL}(u,x) \land \text{RL}(v,y) \land \exists x_1 \exists y_1 (\text{RL}(u,x_1) \land \text{RL}(v,y_1)) \land \forall x_2 (\text{RL}(u,x_2) \rightarrow X(x_1,x_2)) \land \forall y_2 (\text{RL}(v,y_2) \rightarrow X(y_1,y_2)) \land y_1 \leq_i x_1) \lor (\text{end}(x) \land x = y)](x,y) \]

I.e., \( x \preceq_{\text{BI}} y \) whenever \( x \) and \( y \) respectively belong to two ancestor-connected sets of leaves \( X \) and \( Y \) (which least common-ancestor is a turn for player \( i \)) such that the most plausible leaf in \( X \) is better for \( i \) than the most plausible leaf in \( Y \). Recall that \( x \preceq_{\text{BI}} y \) iff whenever \( y \) is reached via a BI move from the least common ancestor of \( x \) and \( y \), then so is \( x \). It follows that the relation defined here is the intended \( \preceq_{\text{BI}} \) relation.

We conclude that the algorithmic analysis of Backward Induction and its procedural doxastic analysis in terms of forming beliefs amount to the same thing. Still, as with the iterated announcement scenario, the iterated upgrade scenario also has some interesting features of its own. One is that, for logicians, it yields fine-structure to the plausibility relations that are usually treated as primitives in models for doxastic logic. Thus games provide an underpinning for possible worlds semantics of belief that seems of interest per se.

6.6 Midway Conclusion: Dynamic Foundations

Extensional equivalence, intensional difference We have now seen how three different approaches to analyzing Backward Induction turn out to amount to the same thing. To us, this means that the notion is stable, and that, in particular, its fixed-point definition can serve as a normal form. This motivates taking a closer look at fixed-point logics for game solution. Of course, as we have observed, extensionally equivalent definitions can still have interesting intensional differences in terms of what they suggest. For instance, we see the above analysis of strategy creation and plausibility change as one more concrete case study for a general conceptual issue: the fact that agents’ beliefs and rational action are deeply entangled in the conceptual foundations of decision and game theory.
Dynamic instead of static foundations for game theory  As we also said already, one key feature of our dynamic announcement and upgrade scenarios is this. In the terms of [18], they are \textit{self-fulfilling}: ending in non-empty largest sub-models where players have \textit{common knowledge} or \textit{common belief of rationality}.\footnote{We forego the issue of logical languages for explicitly \textit{defining} the limit submodel.} \footnote{We also forego the further analysis of the limit behavior of upgrade actions on game models. For general models, [9] finds some curious phenomena, such as plausibility cycles, and they prove a general result stating when at least absolute beliefs stabilize in the limit. There are interesting general issues of fixed-point definability for predicates in limit models of dynamic epistemic procedures, that link up with our analysis. We leave this for further work.} Thus, this dynamic style of game analysis is a big change from the usual static characterizations of Backward Induction in the epistemic foundations of game theory. Common knowledge or belief of rationality is not assumed, but \textit{produced} by the logic.

6.7 Test Case: Variants of Backward Induction

We analyzed our standard BI relation and found a number of things: it is definable in fixed-point logic, it can also be analyzed alternatively as a subset of nodes and as a plausibility relation in tightly correlated ways. But are the preceding results just special effects for the notion of Backward Induction chosen here? As a test case, we will now look how all these themes work for the variant BI’ defined in Section 6.2, where preference between sets of outcomes referred to ensuring a greater minimal value:

\textbf{Definition 6.7.1.} BI’ is the largest subrelation of the \textit{move} relation in a finite game tree satisfying the two properties that (a) the relation has a successor at each non terminal node, and (b) CF1 holds.\footnote{The fact that there is such a relation will follow from theorem 6.7.2.}

\begin{align*}
\text{move}(x, y) \land \bigwedge_i (\text{turn}_i(x) \rightarrow \forall u((\text{end}(u) \land X^*(y, u))) \rightarrow \forall z(\text{move}(x, z) \rightarrow \exists v((\text{end}(v) \land X^*(z, v) \land v \leq_i u))))
\end{align*}

where not all occurrences of the relation symbol $X$ are positive. But there are two alternative candidates for general fixed-point logics that can be used: FO(PFP) and FO(IFP) (see Chapter 2). We will show how to do this.

6.7.1 Defining BI’ in Partial Fixed-Point Logic

For a start it will be easier to compute the fixed-point we are interested in using FO(PFP). In FO(PFP) we usually focus on finite models (an extension of the
framework to infinite structures can be found in [97]) and we can consider fixed-points of arbitrary formulas that are reached by a similar sequence of iteration stages as in the case of FO(LFP). We saw in Chapter 2 that the only difference is that the resulting operator is not necessarily monotone and hence, whenever a fixed-point is reached, it does not necessarily correspond to the least fixed-point of this operator. Additionally, whenever no fixed-point is reached, we simply evaluate the corresponding formula as false.

**Theorem 6.7.2.** The relational BI'-strategy is definable in FO(PFP).

**Proof.** Now we can use CF1 and prefix it with a PFP-operator, or equivalently (as we already noticed that each subrelation of the move relation has a unique corresponding set of nodes), we can put:

\[
\mathrm{BI}'(x, y) = \text{move}(x, y) \land [PFP_{X,y}\exists x(\text{move}(x, y) \land \bigwedge_i (\text{turn}_i(x) \rightarrow \forall z (\text{move}(x, z) \rightarrow \\
\forall u((\text{end}(u) \land [TC_{y,u}\text{move}(y, u) \land X(u)](y, u) \rightarrow \\
\exists v(\text{end}(v) \land [TC_{z,u}\text{move}(z, u) \land X(v)](z, v) \land v \leq_i u)))\] \]

In order to see that this formula defines a unique non-empty relation, let us rewrite the subformula that is inside the fixed-point operator, using only the non-reflexive counterpart \(TC^+\) of the transitive closure operator \(TC\), so that the variable \(X\) appears only in the scope of a \(TC^+\)-operator:

\[
\exists x(\text{move}(x, y) \land \bigwedge_i (\text{turn}_i(x) \rightarrow \forall z (\text{move}(x, z) \rightarrow (\varphi_2 \lor \varphi_4 \lor \varphi_1 \lor \varphi_3)))
\]

with

- \(\varphi_1 := \text{end}(y) \land \text{end}(z) \land z \leq_i y\)
- \(\varphi_2 := \forall u((\text{end}(u) \land [TC_{y,u}^+\text{move}(y, u) \land X(u)](y, u) \rightarrow (\text{end}(z) \land z \leq_i u))\)
- \(\varphi_3 := \text{end}(y) \land \exists v(\text{end}(v) \land [TC_{z,u}^+\text{move}(z, u) \land X(v)](z, v) \land v \leq_i y\)
- \(\varphi_4 := \forall u((\text{end}(u) \land [TC_{y,u}^+\text{move}(y, u)](y, u) \rightarrow \exists v(\text{end}(v) \land [TC_{z,u}^+\text{move}(z, v) \land X(v)](z, v) \land v \leq_i u))\)

The essence of the argument is that at any stage \(k\) of the fixed-point iteration, the computed set stabilizes at points whose siblings are along the child relation at greatest distance \(\leq k\) to a leaf. By “stabilizes at some set of points at stage \(k\)” we mean that for every point in that set, the point belongs to the fixed-point...
approximant at stage \( k \) if and only if it belongs to the fixed-point approximant at every stage greater than \( k \). At the first stage of the fixed-point iteration, the formula \( \varphi_1 \) (which does not contain any occurrence of \( X \)) determines once and for all whether leaves that have only leaves siblings belong to the current and later fixed-point approximants. Then, at stage \( k \) the same is similarly determined for points whose siblings are along the child relation at greatest distance \( \leq k \) to a leaf. This is ensured by the syntactic shape of the formulas \( \varphi_2, \varphi_3 \) and \( \varphi_4 \), in which the variable \( X \) appears only in some restricted “guarded form” inside the formula \([TC_{yu}^+move(y,u) \land X(u)](y,u)\), where \( u \) refers to a point which is strictly lower down \( y \) or inside the formula \([TC_{zu}^+move(z,u) \land X(u)](y,u)\), where \( u \) refers to a point which is strictly lower down a sibling \( z \) of \( y \).

6.7.2 Defining \( \text{BI}' \) in Inflationary Fixed-Point Logic

In the \( \text{BI}' \) partial fixed-point computation of the previous section, nothing was preventing nodes which were ruled out at some stage of the induction process to reappear at a later stage, which could at first sight suggest that the process is not inflationary. But Theorem 6.7.2 can still be refined and \( \text{BI}' \) can be defined in the computationally better-behaved logic \( FO(IFP) \) as an inflationary process.\(^{21}\)\(^{22}\) The trick there is to use a simultaneous fixed-point induction, explicitly using an additional inductive “stable” predicate in order to progressively define the stables elements which can be safely added to the fixed-point at each stage of the computation. Note that this is a general idea (see for instance the use of simultaneous modal fixed-point formulas in [16]). Let us also recall that allowing simultaneous fixed-points does not increase the expressive power of \( FO(IFP) \) (see Chapter 2).

**Theorem 6.7.3.** The relational \( \text{BI}' \)-strategy is definable in \( FO(IFP^1) \).

**Proof.** We use a simultaneous fixed-point formula \([IFP X : S](y)\), where:

\[
S := \begin{cases} 
  Xy & \leftarrow \Phi(X,y) \land Y(x) \\
  Yw & \leftarrow \Psi(Y,w)
\end{cases}
\]

with:

\[
\Phi(X,y) := \exists x(move(x,y) \land \bigwedge_i (turn_i(x) \rightarrow \forall z(move(x,z) \rightarrow \\
\end{cases}
\]

\[\text{This is interesting, since [18] already observed how the limits of iterated public announcement procedures on modal models are definable in FO(IFP), and in fact, usually in the modal inflationary calculus, the extension of the modal \( \mu \)-calculus by means of inflationary fixed-points ([49]).}\]

\[\text{From the preceding fact, we can conclude (using [85], [98]) that there is an equivalent definition for \( \text{BI}' \) in FO(LFP) after all, though the latter may involve extra predicates, with a computation no longer matching the natural stages of our algorithm.}\]
∀u((end(u) ∧ [TC_{yu}move(y, u) ∧ X(u)](y, u) → ∃v(end(v) ∧ [TC_{zu}move(z, u) ∧ X(v)](z, v) ∧ v ≤ i u))\\)

and

Ψ(Y, w) := ∃x(move(x, w) ∧ end(w) ∧ ∀y(move(x, y) → end(y)) ∨ (∀y(move^+(w, y) → Y(y)) ∧ ∃x(move(x, w) ∧ ∀z∀z'(move(x, z) ∧ move^+(z, z') → Y(z'))))

Φ(X, y) is exactly the formula of which we considered the partial fixed-point in the proof of Theorem 6.7.2, whereas we use Ψ(Y, w) in order to ensure that at any stage k of the fixed-point iteration, the fixed-point approximant can only contain points whose siblings are along the child relation at greatest distance ≤ k to a leaf (we noticed in the proof of Theorem 6.7.2 that these points are “stable” at stage k). It follows that this formula is equivalent to the formula in the proof of Theorem 6.7.2. □

As in the case of Bl, now that we have a definition of Bl’, we can apply to it the general syntactic translation in Proposition 6.5.14 and obtain a definition of the associated plausibility order ≤_{Bl’}.

Theorem 6.7.4. ≤_{Bl’} is definable in FO(IFP^1).

Proof. As Bl’ is definable in FO(IFP^1), it follows from Proposition 6.5.14 that:

\[ y ≤_{Bl'} z = ∃x(LCA(x, y, z) ∧ (∃w(Bl'(x, w) ∧ move^*(w, z)) → ∃w'(Bl'(x, w') ∧ move^*(w', y))))) \]

It is time to conclude our analysis. We have shown how Bl’, our pilot example of an alternative game solution procedure, can indeed be defined in fixed-point logics of trees. We were even able to do this in different formalisms. □

6.7.3 Alternative: Recursion on Well-Founded Tree Order

We have now given a definition of Bl’ in standard fixed-point logics over general models. But let us mention that an alternative take is also possible. One major feature of game solution procedures like Backward Induction is their exploiting the inductive structure of extensive games, via the well-founded tree dominance order toward the leaves.\(^{24}\) Such orderings allow for recursive definitions that yield

\(^{23}\)This raises the issue of which fixed-point language is most congenial to analyzing games, something to which we return in later sections.

\(^{24}\)But other recursions are possible, too. Both finite and infinite trees allow for recursive definitions over the well-founded tree order in the opposite past direction toward the root.
uniqueness even without positive occurrence:

**Example 6.7.5** (Fixed-points in modal provability logic ([30])). On finite trees, any modal formula of the form $p \leftrightarrow \varphi(p)$ where $p$ occurs only “guarded” (that is, in the scope of at least one modality) in the formula $\varphi$, defines a unique proposition $p$. One proves this by induction on the well-founded tree order.\(^{25}\)

This includes examples like the following:

**Example 6.7.6** (Broader well-founded recursion). Consider the definition $p \leftrightarrow \neg \Box p$. On a 3-nodes linear order

```
1 2 3
```

starting from any set as a value for $p$, this will stabilize with $p = \{2\}$. But it is easy to see that an inflationary bottom-up procedure for this formula stops in the pre-fixed point $\{1, 2\}$, and the deflationary top-down procedure stops in the post-fixed point $\emptyset$. Neither of these is even a fixed-point. What one can see more precisely in the straightforward approximation procedure, without forcing increasing or decreasing sets, is this: starting the iteration from any initial set will gradually get the predicate right, successively, at all nodes lying at increasing height from the leaves.

Of course, our analysis for Backward Induction did not use these simple modal languages, nor did it just use the simple tree dominance order. Still, by inspection of our earlier formulas and arguments the following result is easily seen to hold.

**Fact 6.7.7.** Stated as an equivalence, the Rationality principles CF1 and CF2 both define a unique subrelation of the move relation by recursion on a well-founded order on the nodes of finite trees: viz. the composition of the relations sibling and dominance.

And this unique relation may also be computed for other versions which like CF1 lack positive syntax but do descend along the well-founded order. One can start with any subrelation of the move relation, and then compute according to the given instruction. At any stage $k$, the range of the fixed-point relation stabilizes at points whose siblings are along the child relation at a greatest distance $\leq k$ to a leaf.

**Example 6.7.8** (Computing a fixed-point for CF1). Consider this game, with values on leaves written as pairs (value for $E$, value for $A$):

\(^{25}\)The result was originally shown for transitive conversely well-founded frames, but the same argument applies to finite trees. Additionally, it is known as the de Jongh Sambien Theorem that such fixed-points are modally definable.
Let $R^0$ be the whole move relation. Then $R^1$ is marked in black below:

This still gets the fixed-point relation wrong at the root, but in the next stage we get the stable solution:

**Remark 6.7.9** (Recursion over other well-founded orders than the sibling-dominance order). Let us note that we could also have shown Theorem 6.7.4 by providing a more direct definition of $\preceq_{BR}$ in $\text{FO(IFP)}$. We think this is interesting, because it gives another example of well-founded order over finite trees than the sibling-dominance order. As in Remark 6.5.17, we use the following shorthand:

$$RL(u, x) \approx Move^*(u, x) \land \text{end}(x)$$

Now we put $x \preceq_{BR} y = [IFP X : S](x, y)$, where:

$$S := \begin{cases} X(x, y) \leftarrow \Phi(X, Y, x, y) \\ Y(w) \leftarrow \Psi(Y, w) \end{cases}$$

with:

$$\Phi(X, x, y) \approx \bigvee_i \exists z \exists u \exists v (\text{Turn}_i(z) \land Y(z) \land Move(z, u) \land Move(z, v) \land$$

$$RL(u, x) \land RL(v, y) \land$$

$$\forall x_1 ((RL(u, x_1) \land \forall x_2 (RL(u, x_2) \rightarrow X(x_1, x_2))) \rightarrow$$

$$\exists y_1 (RL(v, y_1) \land \forall y_2 (RL(v, y_2) \rightarrow X(y_1, y_2)) \land y_1 \leq_i x_1))$$
\[\forall \end{end}(x) \land x = y\]

and

\[\Psi(Y, w) : \approx \end{end}(w) \lor \forall y(Move^+(w, y) \rightarrow Y(y))\]

\(\Phi(X, x, y)\) corresponds to the BI' variant of the formula in Proposition 6.5.16. Still, the different notion of preference encoded here entails that the first occurrence of the variable \(X\) is negative. Hence, as in the case of BI', we need to rely on a trick in order to use an inflationary fixed-point computation. The simultaneous fixed-point formula given above ensures via the formula \(\Psi(Y, w)\) that at any stage \(k\) of the fixed-point iteration, the computed relation stabilizes for couples of leaves which least common ancestor lies at maximal distance \(\leq k\) to a leaf. To illustrate it via a simple example, this amounts to assigning numbers to nodes in the tree as follows:

```
  3
 / \
0   2
 / \
0   1
 / \
0   0
```

Leaves are labeled 0 and nodes that are at a maximal distance \(n\) to a leaf are labeled \(n\), so in particular the root is labelled with the length of the maximal path in the tree. On the tree pictured above for instance, the root is labelled 3. Hence it is only decided at stage 3 of the induction whether the first child of the root (which is a leaf) is more plausible or less plausible than all the other leaves in the tree.

In the next section, we will explore the idea of recursion exploiting the presence of well-founded tree orders further. We will introduce what we call “order-conform” operators, and show how these fit with fixed-point definitions in a natural way. This material forms a digression from the main game-oriented line of this chapter, which will be resumed in Section 6.9.

### 6.8 Excursion: Order-Conform Fixed-Point Logics

The analysis in Section 6.7.3 suggests the introduction of a logical formalism for games that can access the well-founded tree order directly.\(^{26}\) But we also

\(^{26}\) Relevant proposals in the literature include the more general “non monotone inductive definitions” of [54]. Such definitions need not have a fixed-point at all, even though some
believe that a more abstract analysis of conditions under which such recursions are successful could be interesting, not only from a game-theoretical point of view, but also from a more general logical perspective. In this section, we give a few preliminary definitions and easy results in this direction. We first introduce a general notion of “order-conform” operator. Such operators have the particularity to always yield a unique fixed-point. Then, we introduce and study some of the basic properties of the \( \iota \)-calculus, a simple modal fixed-point logic on finite trees which extends the basic modal language with a unique fixed-point construct. The analyses in this section abstract from the later analysis of game solution concepts by first adopting a more general standpoint, where the object of interest are fixed-point operators defined not necessarily over finite game trees, but over some given class of finite structures. Then, as an example of a syntactically well-defined logic allowing only order-conform fixed-point constructs, we turn to the case of the \( \iota \)-calculus, where we restrict attention to simple (i.e., without any preference order over the leaves) finite tree structures.

6.8.1 Order-Conform Operators

We will adopt a wider perspective than before, as we do not restrict attention to finite game trees and we are generally interested in recursion over some given well-founded order (we will mention the idea in connection with infinite games in Section 6.10). We will focus here for simplicity on finite structures. We will also restrict to monadic fixed-points. First of all, we need this very basic notion:

**Definition 6.8.1** (Ordered partition of a finite set). An ordered partition \( \Pi \) of a finite set \( X \) is an ordered sequence

\[
\Pi(X) = X_1 < \ldots < X_k
\]

of non-empty pairwise disjoint subsets of \( X \), whose union is \( X \).

But then, we are not only interested in isolated finite sets, but rather in classes of finite structures and in ways of ordering their domains. We formalize this using the following notion.

**Definition 6.8.2** (Ordered partition mapping). Let \( \mathcal{C} \) be a class of finite structures. We call ordered partition mapping on \( \mathcal{C} \) a mapping \( \mathcal{P} \) which to every structure \( \mathfrak{M} \in \mathcal{C} \) assigns an ordered partition \( \mathcal{P}(\mathfrak{M}) = X_1 < \ldots < X_k \) of \( \text{dom}(\mathfrak{M}) \).

Now we can define abstractly what it means for an operator on the powerset of the domain of structures in \( \mathcal{C} \) to “go along some order”. Whenever we fix a structure and try to compute a fixed-point for such an operator, from whichever set we begin the computation with, the membership of points which are minimal analysis of various conditions under which they correspond to a unique fixed-point is provided.
in the order will always be fixed in the same way at the first stage of the induction. Moreover, this will be done once and for all (i.e., these points will remain either inside or outside the set to be computed). Similarly, the membership of points which are at some level \( n \) in the order will be fixed once and for all at stage \( n \) of the induction.

**Definition 6.8.3** (Order-conform operator). Let \( \mathcal{C} \) be a class of finite structures and \( F \) an operator on the powerset of the domain of the structures in \( \mathcal{C} \) which for every \( \mathcal{M} \in \mathcal{C} \) assigns to each set \( A \subseteq \text{dom}(\mathcal{M}) \) a set \( F(A) \subseteq \text{dom}(\mathcal{M}) \).

Assume \( F \) is such that there is an ordered partition mapping \( \mathcal{P} \) which to each \( \mathcal{M} \in \mathcal{C} \) assigns \( \mathcal{P}(\mathcal{M}) = X_1 < \ldots < X_k \) and for all \( A, A' \subseteq \text{dom}(\mathcal{M}) \), for all \( n \)

\[
F^n(A) \cap \bigcup_{i \leq n} X_i = F^n(A') \cap \bigcup_{i \leq n} X_i
\]

then we call \( F \) an order-conform operator based on \( \mathcal{P} \).

We now have as immediate consequence of our definition.

**Theorem 6.8.4.** Order-conform operators always have a unique fixed-point.

**Proof.** Let \( F_\varphi \) be an order-conform operator based on \( \mathcal{P} \) on some class \( \mathcal{C} \). Now let \( \mathcal{M} \in \mathcal{C} \) and consider \( \mathcal{P}(\mathcal{M}) = X_1 < \ldots < X_n \). It is immediate that \( F^n_\varphi(\emptyset) \) is a fixed-point of \( F_\varphi \). Now let \( Y \) and \( Y' \) be two fixed points of \( F_\varphi \), i.e., \( F_\varphi(Y) = Y \) and \( F_\varphi(Y') = Y' \). As \( F_\varphi \) is an order-conform operator, by definition 6.8.3,

\[
F^n_\varphi(Y) \cap \bigcup_{i \leq n} X_i = F^n_\varphi(Y') \cap \bigcup_{i \leq n} X_i, \text{ i.e., } Y \cap \text{dom}(\mathcal{M}) = Y' \cap \text{dom}(\mathcal{M}), \text{ i.e., } Y = Y'.
\]

Let us now come back to operators that are yield by fixed-point logic formulas. Sometimes, an \( \text{FO(IFP)} \)-formula always has a unique fixed-point on some given class of structures, but we already noted that this fixed-point is generally not definable as its inflationary fixed-point. Still, we can notice that whenever the formula yields an order-conform operator, we can simply define this unique fixed-point in \( \text{FO(PFP)} \) (as \( \text{FO(IFP)} \) formulas are translatable in \( \text{FO(PFP)} \));

**Proposition 6.8.5.** Let \( \Phi(X, x) \) be an \( \text{FO(IFP)} \)-formula and \( F_\Phi \) an order-conform operator on some finite class of structures \( \mathcal{C} \), then the unique fixed-point of \( \Phi \) on \( \mathcal{C} \) is definable in \( \text{FO(PFP)} \).

But jumping to \( \text{FO(PFP)} \) is not very satisfying and we would like to remain within the expressive power of \( \text{FO(IFP)} \). Let us then restrict to cases where the ordered partition mapping is definable in \( \text{FO(IFP)} \).

**Definition 6.8.6** (\( \text{FO(IFP)} \)-definable partition mapping). Let \( \sigma \) be a relational vocabulary, \( \mathcal{C} \) a class of finite \( \sigma \)-structures and \( \mathcal{P} \) an ordered partition mapping on \( \mathcal{C} \) such that there exists a total order \( \leq_\mathcal{P} \) on the domains of the structures in \( \mathcal{C} \) (with associated strict order \( <_\mathcal{P} \)) definable by an \( \text{FO(IFP)} \)-formula in vocabulary \( \sigma \) and such that for every \( \mathcal{M} \in \mathcal{C} \) with \( \mathcal{P}(\mathcal{M}) = X_1 < \ldots < X_k \), for every valuation \( g \) on \( \mathcal{M} \):
Chapter 6. Fixed-point Logics on Finite Extensive Games

- \( X_1 = \{ a | \mathcal{M}, g[a/y] = \neg \exists x x <_p y \} \) is the set of minimal elements in \( <_p \)

- \( \bigcup_{j \leq i+1} X_j = \{ a | \mathcal{M}, g[a/y] = \forall x (x <_p y \rightarrow x \in \bigcup_{j \leq i} X_j) \} \)

then we say that \( \mathcal{P} \) is an \( \text{FO(IFP)} \)-definable ordered partition mapping.

We do not address here the question of whether there are \( \text{FO(IFP)} \)-formulas yielding order-conform operators (on some given class of structures) that are based on non \( \text{FO(IFP)} \)-definable ordered partition mappings and we leave it open for further investigations. We only make the following easy observation.

**Theorem 6.8.7.** Let \( \Phi(X,x) \) be an \( \text{FO(IFP)} \)-formula in vocabulary \( \sigma \) such that \( F_\Phi \) is an \( \text{FO(IFP)} \) order-conform operator on some class of finite \( \sigma \)-structures \( \mathcal{C} \).

Assume the ordered partition mapping on which \( F_\Phi \) is based is \( \text{FO(IFP)} \)-definable.

Then there exists an \( \text{FO(IFP)} \)-formula \( \Psi(x) \) such that

\[
[\text{PFP}_x \Phi(X,x)^{\text{FO(PFP)}}](x) \leftrightarrow \Psi(x)
\]

(where \( \Phi(X,x)^{\text{FO(PFP)}} \) is the translation of \( \Phi \) in \( \text{FO(PFP)} \))

**Proof.** The idea is similar to the one in the proof of Theorem 6.7.3 and \( \Psi(x) \) can be written as the following simultaneous fixed-point formula:

\[
[\text{IFP} X : S](x)
\]

with

\[
S := \begin{cases} 
X x \leftarrow \Phi(X,x) \land Y(x) \\
Y y \leftarrow (\neg \exists x x <_p y) \lor \forall x (x <_p y \rightarrow Y(x))
\end{cases}
\]

In the system \( S \), we use the formula \((\neg \exists x x <_p y) \lor \forall x (x <_p y \rightarrow Y(x))\) in order to ensure that at any stage \( n \) of the fixed-point iteration, the fixed-point approximant of \( [\text{IFP} X : S](x) \) can only contain points which are in \( \bigcup_{i \leq n} X_i \) where \( X_i \) is the \( i^{th} \) set in the ordered sequence \( \mathcal{P}(\mathcal{M}) = X_1 < \ldots < X_k \).

Now Theorem 6.8.7 gives a clean recipe to characterize non-positive variants of \( \text{Bi} \) in \( \text{FO(IFP)} \): it is enough to mimic the simultaneous fixed-point system in the proof of Theorem 6.7.3. Only the way we define the set \( X \) will vary, but we will keep the same formula to define the set \( Y \), as the recursion will always occurs along what we earlier identified as the “sibling dominance” order in the tree. Note that other game-theoretic notions might also be definable in \( \text{FO(IFP)} \) using the same idea, provided that the computation from which they arise be similarly based on some \( \text{FO(IFP)} \)-definable order on the nodes of the tree.

One further question is whether we could find natural syntactic characterizations of \( \text{FO(IFP)} \)-formulas yielding order-conform operators, for instance in the case of finite trees. The proof of Theorem 6.7.2 suggests a few patterns, but for now we will leave the question open. Rather, we will temporarily forget about
the player’s preferences labeling the leaves of finite game trees and we will turn to the case of modal fixed-point logics on plain finite trees. We view this excursion as a first step towards an understanding of the underlying action structure of prospective more elaborate fixed-point logics of finite game trees.

6.8.2 The Modal $\nu$-Calculus

To show how order-conform formalisms make sense, we will briefly develop one particular modal fixed-point logic over finite Kripke models based on finite tree frames. While this system is clearly not rich enough to define the game solution algorithms we had earlier, it serves as a nice pilot example of what might be achieved with simple logical languages that exploit well-founded tree orders. We call this logic the $\nu$-calculus ($L_\nu$), by reference to Russell’s definite description operator $\nu x$, which is to be read as “the unique $x$ such that”. $L_\nu$ is a simple modal logic on finite trees which extends the basic modal language with a fixed-point operator which can be applied to formulas that are not necessarily positive but which satisfy a syntactic condition of “guardedness” which ensures that they yield order-conform operators. We show in particular that this logic has exactly the same expressive power as the modal $\mu$-calculus on finite trees. We will restrict here to the class of finite trees Kripke-structures, i.e., finite Kripke structures based on a frame which belongs to the class of finite tree frames $T$. The question of the $\nu$-calculus was first raised to us by Johan van Benthem.

**Definition 6.8.8 ($L_\nu$).** Let $\sigma$ be a propositional signature, and let $V = \{x_1, x_2, \ldots\}$ be a disjoint countably infinite stock of *propositional variables*. The set of $L_\nu$-formulas in vocabulary $\sigma$ is generated by the following inductive definition:

$$\varphi, \psi, \xi := At \mid \top \mid \neg \varphi \mid \varphi \lor \psi \mid \varphi \land \psi \mid \varphi \rightarrow \psi \mid \lozenge \varphi \mid \nu x_1. \xi$$

where $At \in \sigma \cup V$ and in the last clause, $x_1$ occurs only guarded in $\xi$ (i.e., within the scope of a $\lozenge$-operator). The satisfaction relation is inductively defined between $L_\nu$-formulas and pointed structures $(T, V, w)$ where $T \in T$ as in the case of the $\mu$-calculus using an auxiliary assignment to interpret formulas with free variables. The only difference concerns the $\nu$-operator and we interpret $\nu x. \varphi$ as the unique fixed-point of the operator $F_\varphi$.

The semantics is consistent, because of the following:

**Lemma 6.8.9.** Let $\varphi(x)$ be a $L_\nu$-formula in which $x$ is guarded, then $\varphi(x)$ has a unique fixed-point.

**Proof.** The proof is by induction on the number of nested fixed-point operators in $\varphi(x)$. We will show that $F_\varphi$ is an order-conform operator on finite trees based on the following ordered partition mapping of the set of nodes:
\( x \in X_{i+1} \) iff \( x \) lies at maximal distance \( i \) to a leaf.

The lemma will follow by Theorem 6.8.4. For the base case, let us assume \( \varphi(x) \) is purely modal (i.e., does not contain any \( \iota \)-operator). Let \( \mathcal{M} \) be a finite tree Kripke structure, we will show by induction on \( n \) that for all \( A, A' \subseteq \text{dom}(\mathcal{M}) \),

\[
F^n_\varphi(A) \cap \bigcup_{i \leq n} X_i = F^n_\varphi(A') \cap \bigcup_{i \leq n} X_i
\]

The base case is \( n = 1 \), where \( X_1 \) is the set of all leaves in the tree (i.e. nodes which lie at maximal distance 0 to a leaf). As the variable \( x \) in \( \varphi \) is guarded, it occurs only in the scope of at least one \( \lozenge \)-operator. Let \( a \in X_1 \), the fact that \( a \in F_\varphi(A) \) or \( a \in F_\varphi(A') \) depends only on the fact that points that are lower down \( a \) in the tree belong to \( A \) or \( A' \) respectively, but points in \( X_1 \) being only leaves, there are no such points. Hence \( a \in F_\varphi(A) \) if and only \( a \in F_\varphi(A') \) and:

\[
F_\varphi(A) \cap X_1 = F_\varphi(A') \cap X_1
\]

Now assume the property holds for \( n \), then the property follows for \( n + 1 \) by a similar argument as in the base case. Let \( a \in \bigcup_{i \leq n+1} X_i \), as the variable \( x \) in \( \varphi \) is guarded, the fact that \( a \in F^{n+1}_\varphi(A) \) or \( a \in F^{n+1}_\varphi(A') \) depends only on the fact that points that are lower down \( a \) in the tree belong to \( F^n_\varphi(A) \) or \( F^n_\varphi(A') \) respectively. By induction hypothesis, for every such point \( b, b \in F^n_\varphi(A) \cap \bigcup_{i \leq n} X_i \) if and only if \( b \in F^n_\varphi(A') \cap \bigcup_{i \leq n} X_i \). Hence \( a \in F^{n+1}_\varphi(A) \) if and only if \( a \in F^{n+1}_\varphi(A') \) and:

\[
F^{n+1}_\varphi(A) \cap \bigcup_{i \leq n+1} X_i = F^{n+1}_\varphi(A') \cap \bigcup_{i \leq n+1} X_i
\]

The inductive case for more complex formulas \( \varphi(x) \) containing nested \( \iota \)-operators is similar.

Let us point out here that the argument used in the proof of Lemma 6.8.9 is similar to the one used in order to show the fixed-point lemma of provability logic (see [30]), recalled in Example 6.7.5. Note that provability logic interprets basic modal formulas on Kripke structures based on “transitive conversely well-founded frames”, whereas we considered here finite trees. But we can simulate formulas of provability logic on finite trees in \( L_\iota \) by considering the transitive closure of the \( \Box \)-operator of \( L_\iota \). Hence the fact recalled in Example 6.7.5 also follows from Lemma 6.8.9.

Now let us recall a result which immediately implies that the \( \iota \)-calculus extends the \( \mu \)-calculus on finite trees. We first need the general notion of guardedness for a modal fixed-point formula (by modal fixed-point formula we mean a formula in a logic extending basic modal logic with a fixed-point operator):
Definition 6.8.10. A modal fixed-point formula is guarded, if all propositional variables that are bound by and thus occur in the scope of a fixed-point operator are also in the scope of a modality that is itself in the scope of the fixed-point operator.

Proposition 6.8.11 ([99]). Every formula of the \( \mu \)-calculus is equivalent to a guarded formula.

We will now show that \( L_\iota \) and \( L_\mu \) have exactly the same expressive power on finite trees. Moreover, we show that for every \( L_\iota \)-formula there is an effective procedure which computes a \( L_\mu \)-formula which is equivalent on finite trees, and vice versa.

Theorem 6.8.12. The \( \iota \)-calculus and the \( \mu \)-calculus on finite trees are effectively equi-expressive.

Proof. The fact that the \( \iota \)-calculus extends the \( \mu \)-calculus on finite trees follows from Proposition 6.8.11 because the \( \iota \)-operator being a unique fixed-point operator, it can equivalently be replaced by a \( \mu \)-operator whenever it is prefixed to a positive formula (whenever a formula has a unique fixed-point, then this fixed-point is also its least fixed-point). Hence, for every guarded formula \( \varphi \) of the \( \mu \)-calculus there exists a formula of the \( \iota \)-calculus which is equivalent to \( \varphi \) on finite trees (simply replace every \( \mu \)-operator in \( \varphi \) by a \( \iota \)-operator). But using the Janin Walukiewicz Theorem ([91], see Chapter 2), one can even refine this inclusion. It is immediate that on finite trees the \( \iota \)-calculus is contained in the partial iteration calculus MPC (see Chapter 2), which is bisimulation-invariant (see [97] and Chapter 2), so it is also bisimulation-invariant. Note that the bisimulation-invariance of \( L_\iota \) also follows from the fact that there is a recursive procedure which transforms any \( L_\iota \)-formula \( \varphi \) into a MIC-formula \( \varphi_{\text{MIC}} \) which is equivalent to \( \varphi \) on finite trees. We can define the procedure by induction on the complexity of \( \varphi \). The only interesting clause is \( \iota x.\varphi_{\text{MIC}}(x) := (if \ p \ x : S) \) where:

\[
S := \begin{cases} 
  x \leftarrow \varphi_{\text{MIC}}(x) \land y \\
  y \leftarrow \Box \bot \lor \Box y
\end{cases}
\]

Moreover, the \( \iota \)-calculus can be embedded in MSO. This can be shown by induction on the complexity of \( L_\iota \)-formulas using a standard translation argument (see Chapter 2) where the new clause is \( ST_x(\iota x.\varphi(x_i)) = \exists X_i( (\forall y(ST_y(\varphi(x_i)) \leftrightarrow X_i(y)) \land X_i(x_i)) \). The result then follows by the Janin Walukiewicz Theorem (see Theorem 2.2.13 in Chapter 2), which says that, in particular on finite trees, every bisimulation-invariant logic which is contained in MSO is also contained in the \( \mu \)-calculus. Moreover, from the fact that the translation from bisimulation-invariant MSO-formulas to \( L_\mu \)-formulas given by Theorem 2.2.13 is effective, it follows that the translation from \( L_\iota \)-formulas to \( L_\mu \)-formulas is effective. \( \square \)
Corollary 6.8.13. $L_\iota$ is decidable.

Proof. To determine whether a $L_\iota$-formula is satisfied in a finite tree model, first translate it in MSO, then construct an equivalent finite tree automata and check for emptiness of the language of the automata. For details on the relation in between finite tree automata and MSO-formulas on finite trees, see [127].

Let us now list a few questions. First, it would be interesting to look at the details of the translation procedure obtained in Theorem 6.8.12 via the Janin Walukiewicz Theorem. As it involves translations in MSO, it is likely that there might be more direct and efficient translation procedures. This question is closely related to the issue of the possible greater succinctness of $L_\iota$ as compared to the $\mu$-calculus on finite trees\footnote{For the notion of compared succinctness of logics which have the same expressive power on a given class of structures, in particular on finite trees, we refer to [79] and [78].}, which we leave here as an open problem. Another question is the following. Consideration of the proof used in [40] in order to show the completeness of the $\mu$-calculus on finite trees strongly suggests that a complete axiomatization of $L_\iota$ could easily be obtainable using a similar method. We leave the details of this question for further work. Then, it would also be interesting to determine more precisely general syntactic conditions satisfied by MIC formulas for which there is a $L_\iota$-formula which is equivalent on finite trees. Such a criteria would allow to isolate a fragment of MIC which would be decidable on finite trees (remember that we noted in Chapter 2 that MIC is undecidable already on finite words). Also note that the decidability of $L_\iota$ immediately implies that the set of guarded MPC-formulas is decidable on finite trees.

Finally, let us conclude this section by saying that we hope our analysis of game solution concepts can also feed back to general fixed-point logics by raising logical questions that are of interest per se. One such question concerns logics featuring recursion along a well-founded order definable in structures of some given class; as a starting point for exploring the direction, we believe the simple modal $\iota$-calculus introduced in the present section would deserve to be explored further.

6.9 Towards Well-Behaved Fixed-Point Logics on Finite Extensive Games

After our excursions into order-conform fixed-point logics on trees, we return to the main line of this chapter: defining solution concepts for games in the sense of game theory. Actually, we will not develop further theory here, but confine ourselves to raising a few key questions which arise at once.
Finding suitable fragments  Of course, game solution procedures need not use the full power of logical languages that can define recursive procedures. Thus, there is a question which fragments are needed in our analysis. It might make sense to look at decidable fragments such as the modal $\mu$-calculus - and indeed, [18] points out how the latter suffices, e.g., for defining the game solutions needed for Zermelo’s Theorem. This may look too poor, since we often want to define relations on trees, and not just unary predicates. But we have already seen how subsets of the $\text{move}$ relation are encoded by unary predicates, so a lot can be done in this way. Still, the intriguing issue is this. Crucially, game solution intertwines two different relations on trees: the $\text{move}$ relation and the preference relations for players on endpoints. And the question is what happens to the known properties of computational logics when we add such preference relations. In particular, the following intriguing issue then arises.

Potential problem: the complexity of rationality  In logics of action and knowledge, it is well-known that apparently harmless assumptions such as Perfect Recall for agents make the bimodal logic undecidable, and sometimes even $\Pi_1^1$-complete ([86]). The reason is that these assumptions generate commuting diagrams for actions $\text{move}$ and epistemic uncertainty $\sim$ satisfying a “confluence” property of the form

$$\forall x \forall y ((\text{move}(x, y) \land y \sim z) \rightarrow \exists u (x \sim u \land \text{move}(u, z))$$

that can serve as the basic grid cells in encodings of Tiling Problems in the logic. Thus, the logical theory of games with players that have perfect memory is more complex than that of forgetful agents ([24]).

But now consider the non-epistemic property of Rationality that mixes action and preference. The earlier properties $\text{CF1}$, $\text{CF2}$ have a similar flavor: they express the existence of a confluence diagram involving action and preference links. For instance, $\text{CF1}$ said this:

$$\forall x \forall y ((\text{turn}_i(x) \land \sigma(x, y)) \rightarrow \forall z (\text{move}(x, z))$$

$$\rightarrow \forall u ((\text{end}(u) \land \sigma^*(y, u)) \rightarrow \exists v (\text{end}(v) \land \sigma^*(z, v) \land v \leq_i u))))$$

So, what is the complexity of fixed-point logics for players with this kind of behavior? Can it be that Rationality, a widely used property meant to make behavior simple and predictable, actually makes its logical theory complex? Concrete instances of this open problem arise once we fix a sufficiently expressive logical language over trees.\(^{28}\)

\(^{28}\)Model-checking complexity and definability on finite trees. Balder ten Cate has reminded us of the potential use of descriptive complexity theory ([90]) for studying finite games. First, checking for game solutions is related to model checking logical formulas, say, stating the intended effects of players strategies. As an example, since both $\text{FO}(\text{LFP})$ and $\text{FO}(\text{IFP})$ capture $\text{PTIME}$ on finite models (given an enumeration order on the tree), it should be close to defining
The Gist of it all: Modal Logics of Best Action  We have made a plea for analyzing game solution procedures explicitly in rich logics. This follows the program of making strategies explicit advocated in [19]. But while this is useful in some cases, there is also the opposite direction of judiciously hiding information about the machinery of strategies when it is not needed. In practical reasoning, we are often only interested in our best actions without all details of their justification. Game solution procedures take a model with actions and preferences, and then compute a new relation of best action. As a mathematical abstraction, it would be good to extract a simple surface logic (a small modal fragment of complex fixed-point logics) for reasoning with best actions, while hiding most of the machinery:

Open problem  Can we axiomatize the modal logic of finite game trees with a move relation and its transitive closure, turns and preference relations for players, and a new relation best as computed by Backward Induction?

We conjecture that we get a simple modal logic for the moves (these exist) plus a basic preference logic, while the modality \( \langle \text{best} \rangle \) satisfies some obvious base laws plus one major bridge axiom that we already encountered earlier:\(^{29}\)

Fact 6.9.1. The following modal axiom corresponds to CF2 by standard techniques:

\[
(turn_i \land \langle \text{best} \rangle \langle \text{best} \rangle^*(\text{end} \rightarrow p)) \rightarrow [\text{move}](\langle \text{best} \rangle^*(\text{end} \land \langle \text{pref}_i \rangle p)
\]

In this concrete setting, the earlier problem returns that the Rationality assumption built into this logic may be a grid property leading to undecidability. Is the modal logic of best action decidable?\(^{30}\)

6.10 Further Issues in Extended Game Logics

In addition to the definability issues that we solved so far, game logics raise some other questions and there are many further lines for investigation following up on our stray observations. For instance, we want a more general view of possible all “testable” properties of games. And other results in descriptive complexity theory may be game-theoretically relevant as well.

\(^{29}\) [113] has some related thoughts on “logics of solved games”.

\(^{30}\) In our earlier analysis of Backward Induction, we look at either shrinking game trees, or smaller sets of “best moves” as the recursive procedure unfolds. This suggest that the logic of best moves may need a further modality: not just the above “absolute best” given the game as it is, but also “relative best” given some constraint on the set of nodes or moves considered. (This is similar to having conditional belief as a basic notion in doxastic logics.) If this is right, we may need in order to get completeness more operators than just the one shown in Fact 6.9.1.
representation languages, and on the notions of set preference that determine the dominance relation defining rationality. More generally, it would be of interest to connect our style of analysis for game solution more systematically with that found in epistemic game theory (c.f. [115], [52], [137]), where epistemic models are added describing what players know or believe about the course of the game.

In addition, some extensions to the games themselves seem natural:

**Infinite games** Can we extend our analysis to deal with *infinite games*? A transition to infinite ordinal sequences is easy to add to our iterated announcement or upgrade scenarios. Also, our general fixed-point definitions still make sense in this setting, though the special recursion over a well-founded tree dominance relation is no longer available. But there may be more to this generalization. Typically, in infinite trees, the reasoning changes direction, from “backward” to “forward”. Here is an illustration:

**Example 6.10.1** (Weak Determinacy). The following principle holds in all infinite game trees, for any condition $\varphi$ on histories:

If player $E$ has no strategy forcing $\varphi$ at some stage $s$ of the game, then $A$ has a strategy for achieving a set of runs from $s$ during all of which $E$ never has a strategy forcing $\varphi$ for the remaining game from then on.

In the notation of temporal game logics with forcing modalities $\{\}$, this says

$$\{E\} \varphi \lor \{A\} G \neg \{E\} \varphi$$

Here the reasoning is a typical inverse of Backward Induction. Suppose that $\neg \{E\} \varphi$. $A$’s strategy then arises as follows. If $E$ is to move, then no successor available to her can guarantee a win, since she has no winning strategy now - and so $A$ can just “wait and see”. If $A$ is to move, then there must be at least one possible move leading to a state where $E$ has no winning strategy: otherwise, $E$ has a winning strategy right now after all. Continuing this way, $A$ is bound to produce runs of the kind described.$^{31}$

How would our earlier analysis extend to a setting like this, where infinite histories themselves are the outcomes of the game, and players try to achieve global properties of these?

$^{31}$This argument has a *co-algebraic* flavor, cf. [132], that we do not pursue here.
**Dynamics in games with imperfect information** Moreover, many if not most games have *imperfect information*, with uncertainties for players where they are in the game tree. Think of card games, or other games with restricted observation. Can’t our analysis be extended to this area, where in general, Backward Induction no longer works? We merely illustrate the task ahead with two simple scenarios for the reader to ponder:

**Example 6.10.2** (Strategic reasoning in imperfect information games). In the following games, outcome values are written in the order (“A-value, E-value”):

```
\[
\begin{array}{ll}
A & E \\
E & E & E & E \\
1, 1 & 0, 0 & 0, 1 & 2, 0 \\
3, 2 & 1, 0 & 0, 1 & 2, 2 \\
\end{array}
\]
```

The game to the left seems a straightforward extension of techniques for removing dominated moves, but that to the right raises tricky issues of what A would be telling E by moving right. We leave the question what should or will happen in both games to the reader: [52], [137] have more discussion.

**Language design and game equivalence** As a final perspective, we mention that the choice of a best language for games is also correlated with the choice of an optimal notion of *structural equivalence* between games ([13]). The richer the equivalence, the stronger the language needed to capture its invariant properties. The options for languages that we have discussed here may also reflect the fact that there is no consensus yet on what such a structural notion of game equivalence should be. 34

### 6.11 Coda: Alternatives to Backward Induction and True Game Dynamics

Our discussion in this chapter is basically complete, but we feel that we should mention one more issue that has received quite some attention in the literature

---

32 The tree to the right is adapted from an example in an invited lecture by Robert Stalnaker at the Gloriclass Farewell Event, ILLC Amsterdam, January 2010.

33 Further challenges to our analysis include equilibria with *coalitions* of players, and *simultaneous moves*. Nothing in our logics prevent this; it has just not been done yet.

34 [13] proposes notions of game bisimulation where back and forth moves occur only when there is a switch of turns between players. This seems similar to the notion of “stuttering” encountered in Chapter 4 and 5, but we leave this analogy to further research.
6.11. Coda: Alternatives to Backward Induction and True Game Dynamics

(cf. [25]). It has been claimed that the very reasoning underlying Backward Induction, our hitherto unproblematic running example of a game solution method, is incoherent:

The paradox of Backward Induction

Example 6.11.1 (The “Paradox of Backward Induction”). Recall the style of reasoning toward a Backward Induction solution, as in:

Backward Induction tells us that $A$ will go left at the start, on the basis of logical reasoning that is available to both players. But then, if $A$ plays right (see the black line) what should $E$ conclude? Does not this mean that $A$ is not following the BI reasoning, and that all bets are off as to what he will do later on in the game? It seems that the very basis for the above computation collapses.

Responses to this conceptual difficulty vary - and many authors doubt that there is a genuine paradox here. The characterization result of [6] assumes that players know that rationality prevails throughout, something that [119] calls “rationality no matter what”, a stubborn unshakable belief that players will act rationally later on, even if they have not done so up until now.

[10] essentially take the same tack, deriving the BI strategy from an assumption of “stable true belief” in Rationality, a gentler form of stubbornness stated in terms of dynamic-epistemic logic.

Logics of actions, preference, and agent types  Personally, we are more inclined toward another analysis, in line with [125]. A richer game analysis should add an account of the types of agent that play a game. In particular, we need to represent the belief revision policies by the players, that determine what they will do when making a surprising observation contradicting their beliefs in the course of a game. There are many different options for such policies in the above example, such as

- “It was just an error, and $A$ will go back to being rational”,

---

$^{35}$One can defend this by assuming that the other player only makes isolated “mistakes”.
Our logical analysis so far omits this type of information about players of the game, since our algorithms make implicit uniform assumptions about what they are going to do as the game proceeds.

Belief revision policies are not an explicit part of our models so far. Thus, our fixed-point logics tell only a limited story. Eventually, we may need a richer mathematical model for game solution, that can also deal with the dynamics of how players update knowledge and revise beliefs as a game proceeds.

6.12 Conclusion

We have shown how standard logical fixed-point languages can define game solution procedures and their resulting relations of “best action”. We think that this is a good format for more general studies of game-theoretic notions, including finding alternatives to currently received views. But also, we hope to have shown that the game arena poses interesting problems for existing logics of computation, as one adds further structure that is typical for agents: preference, information, and eventually, even “processing types” for agents. All these contacts may eventually lead to legitimate children of logic and game theory. The chapter has mainly analyzed Backward Induction as a key to seeing how fixed-point logics can be used for game solution. From a logical perspective, the issue is now how to continue, what are the most useful fixed-point logics in the wide array that we have brought to bear and how expressive would be interesting well-behaved (in particular, axiomatizable) fragments of such logics. At the moment, we find it hard to choose, though we do think that both general fixed-point logics and special logics exploiting well-founded relations in trees make sense. We intend to investigate further game solution concepts to get a better sense which of the logics used in this chapter will stand the test of game-theoretic generalization.

\footnote{One reaction to these surprise events might even be a switch to an entirely new reasoning style about the game. That might require more finely-grained syntax-based views of revision.}