Chapter 1 SEMANTICAL EVALUATION GAMES

Any language acquires meaning through semantic interpretation of its expressions. Logical languages usually have a model-theoretic semantics defining when a formula ϕ of the language is true in a model $\boldsymbol{M}$, perhaps with some auxiliary setting. The paradigm for this is the semantics of first-order logic, with its notion

$\boldsymbol{M}, s \models \phi$ with s an assignment of objects in $\boldsymbol{M}$ to variables

Now, stepwise evaluation of first-order assertions can be understood dynamically as a game of evaluation for two players. *Verifier* claims that ϕ is true in the setting $\boldsymbol{M}, s$, *Falsifier* that it is false. This is our most basic logic game. In this chapter we

- explain the practical workings of first-order evaluation games
- establish their adequacy with respect to truth and falsity
- show and explore their more general game-theoretic character
- demonstrate how other logics can be 'gamified' in the same style
- and identify some general issues of 'game logic' behind them including the role of players' strategies and game operations.

Games like this were proposed by Jaakko Hintikka around 1960 – with a grander program in the pioneering monograph Hintikka 1973. Since then, evaluation games have been defined for many languages, both in logic and in computer science. More on the linguistic side, 'game-theoretical semantics' has become an approach of its own in the semantics of natural languages, witness Hintikka & Sandu 1997. Our emphasis here will be on game-theoretic aspects, rather than these applications.

1.1 Evaluation games for predicate logic

Let two parties disagree about a statement ϕ in some situation $\boldsymbol{M}, s$: *Verifier* $\boldsymbol{V}$ claims it is true, *Falsifier* $\boldsymbol{F}$ that it is false. Here are the natural moves of defense and attack in the first-order evaluation game, which we will indicate henceforth as

$\boldsymbol{game}(\phi, \boldsymbol{M}, s)$

Definition 1.1 Moves in evaluation games

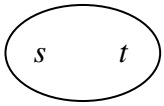
The moves of evaluation games follow the inductive construction of formulas. They involve some typical notions in the dynamics of games, such as choice, switch, and continuation, coming in dual pairs with both players allowed the inititiative once:

atoms *Pd, Rde,...*	***V*** wins if the atom is true, ***F*** if it is false
disjunction $\phi \vee \psi$	***V*** *chooses* which disjunct to play
conjunction $\phi \wedge \psi$	***F*** *chooses* which conjunct to play
negation $\neg \phi$	*role switch* between the two players, play continues with respect to ϕ

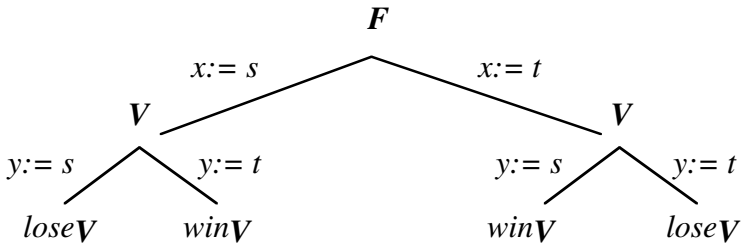
Next, the quantifiers make players look inside ***M***'s domain of objects:

existential quantifiers $\exists x\, \phi(x)$	***V*** *picks an object d*, and then play continues with respect to $\phi(d)$
universal quantifiers $\forall x\, \phi(x)$	the same move, but now for ***F***

The schedule of the game is determined by the form of the statement ϕ. To see this, recall the example from Section I.2.4. Consider a model ***M*** with two objects:



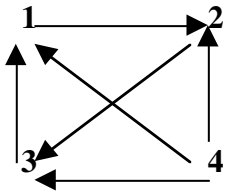
Here is the complete game for the first-order formula $\forall x\, \exists y\, x \neq y$, pictured as a tree of possible moves, with the scheduling read from top to bottom:



We interpret this as a game of perfect information: players know throughout what has happened. Falsifier starts, Verifier must respond. There are 4 possible plays, with 2 wins for each player. But Verifier is the player with a *winning strategy*, in an obvious sense. Strictly speaking, trees like this are not a formal definition of the game yet – but for many purposes, we are better off without further formalities.

Evaluation games for more complex formulas in more structured models can be quite attractive in teaching elementary logic. Here is a simple illustration.

Example 1. 2 Find non-communicators!
Consider the following communication network with arrows for directed links, and with all 'self-loops' present but suppressed in the drawing:



In this setting, the predicate-logical formula $\forall x\ \forall y\ (Rxy \vee \exists z(Rxz\ \&\ Rzy))$ claims that every two nodes in this network can communicate in at most two steps. Here is a possible run of the corresponding evaluation game:

player	*move*	*next formula*
F	picks ***2***	$\forall y\ (R\mathbf{2}y \vee \exists z(R\mathbf{2}z\ \&\ Rzy))$
F	picks ***1***	$R\mathbf{21} \vee \exists z(R\mathbf{2}z\ \&\ Rz\mathbf{1})$
V	chooses	$\exists z(R\mathbf{2}z\ \&\ Rz\mathbf{1})$
V	picks ***4***	$R\mathbf{24}\ \&\ R\mathbf{41}$
F	chooses	$R\mathbf{41}$
test:	***V*** wins	

In this run, Falsifier started off with a threat by picking object ***2***, but then became generous towards Verifier, picking object ***1***. Verifier accepted the present by choosing the true right conjunct, but then tripped up by picking the wrong witness ***4*** instead of ***3***. But once again, Falsifier did not exploit this, by choosing the true right-hand conjunct. Obviously, Falsifier has a *winning strategy* in this game, exploiting the 'counter-example' of object ***2***, which cannot reach ***1*** in ≤2 steps. He even has more than one such strategy, as x=***2***, y=***4*** would also serve.

If you want to play this particular type of game with your friends, you might exploit the following fact. Every finite network in which distinct points always have at least one directed link contains a 'Great Communicator': an object which can reach every other node in at most 2 steps. Game dynamics even has clear educational appeal, witness the software supporting Barwise & Etchemendy 1999.

Note that a formula ϕ itself is a sort of abstract 'game form' which may be pictured as a tree of definite depth but indefinite branching width, indicating a schedule of turns and moves. It becomes a real game only when a particular model ***M*** is chosen,

which specifies the possible quantifier moves and the outcomes for atomic tests – while in general an assignment *s* needs to take care of the free variables in ϕ.

1.2 Truth and Verifier's winning strategies

In our first example, participants were not evenly matched. Player ***V*** can always win: after all, she is defending the truth of the matter. More precisely, she has a *winning strategy*. As in our Introduction, such a strategy is a map from ***V***'s turns to moves following which guarantees, against any counterplay by ***F***, that the game will end in outcomes that are won for ***V***. Clearly, ***F*** has no winning strategy, as this would contradict ***V***'s having one. (Playing two winning strategies against each other results in a contradiction concerning the unique outcome.) Neither does have ***F*** the opposite power of a 'losing strategy': he cannot force ***V*** to finish him off. Thus, players' powers of controlling outcomes may be quite different. Here is the fundamental connection for evaluation games:

Lemma 1. 3 Success Lemma
The following are equivalent for all ***M***, *s,* and first-order ϕ:

(a) ***M***, *s* $\models \phi$

(b) ***V*** has a winning strategy in ***game***(ϕ, ***M***, *s*).

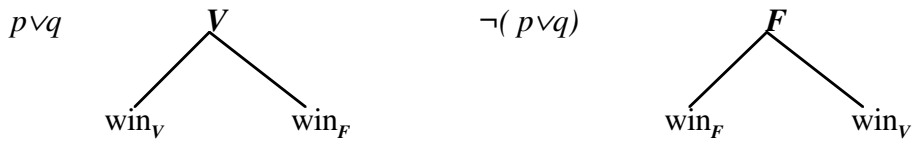
Proof The proof is a direct induction on formulas, One shows simultaneously:

If a formula ϕ is true in (***M***, *s*), then Verifier has a winning strategy,
If a formula ϕ is false in (***M***, *s*), then Falsifier has a winning strategy.

The proof steps exhibit the close analogy between logical operators and ways of combining strategies – and it is also virtually the argument for Zermelo's Theorem (Section I.3.1). We mention just some typical cases. (a) If $\phi \vee \psi$ is true, then at least one of ϕ, ψ is true – say, ϕ. By the inductive hypothesis then, ***V*** has a winning strategy σ for ϕ. But then she has a winning strategy for the game $\phi \vee \psi$: its first move is 'left', after which the rest is the strategy σ. (b) If $\phi \vee \psi$ is false, then both ϕ, ψ are false, and hence by the inductive hypothesis, ***F*** has winning strategies σ, τ for ϕ, ψ, respectively. But then the combination of an initial 'wait-and-see' plus these two is a winning strategy for ***F*** in the whole game $\phi \vee \psi$: If ***V*** goes left in the first move, then ***F*** should play σ, while if ***V*** goes right, ***F*** should play strategy τ. (c) If the formula ϕ is a negation $\neg\psi$, we must use the definition of a role switch. Here is an illustration, before we give the formal argument.

Example Role Switch

Consider the game for a formula $p \vee q$ in a model where p is true and q is false, as well as its dual game $\neg(p \vee q)$, which switches all turns and win markings:



Note the second game works out to that for the De Morgan equivalent $\neg p \vee \neg q$. As this picture illustrates, strategies for ***V*** in a game for $\neg\psi$ are strategies for ***F*** in the game for ψ, and vice versa. Together with the win/lose switch, this underlies our proof of case (c). Suppose that $\neg\psi$ is true. Then ψ is false, and by the inductive hypothesis, ***F*** has a winning strategy in the ψ–game forcing an outcome in the set of ***F***'s winning positions. But this *is* a strategy for ***V*** in the $\neg\psi$–game, and indeed one forcing a set of winning positions for ***V***. The other direction is similar. ■

This is our first connection between a key notion in logic ('truth') and one in game theory ('strategy'). We will broaden this interface as we continue. Admittedly, critics see the Success Lemma as showing how games yield nothing new. If it is a criterion of adequacy, a game-theoretic analysis is 'good' only if it captures some pre-existing logical notion... In these lectures, our focus will be the opposite: what new themes are intrinsic to games, and might change the agenda of classical logic?

1.3 Taking the games seriously

As a first exploration, here are a few comments on the Success Lemma just proved. Simple as it is, there is more to it than meets the eye. In particular, we get some new perspectives on what makes standard predicate logic tick. Many technical distinctions formulated in our discussion will recur in subsequent chapters.

Different winning strategies Truth occurs if and only if there exists a winning strategy for player ***V***, and likewise for falsity and ***F***. But the notion of a winning strategy is more refined – as there can be *more* than one such strategy! E.g., ***F*** had two winning strategies in Example I.2, representing two different counter-examples to the claim. Thus, winning strategies are more refined semantic objects than truth values, which we might call 'reasons' for truth or falsity. This notion does not occur in standard first-order semantics – but it seems a natural addition. Here is an

illustration of the multiplicity of possible reasons for truth or falsity of assertions, as compared with just an all-or-nothing judgment "true" or "false".

Example 1.4 Counting strategies

One can count the maximal number of verifying (falsifying) strategies, say *#(V, ϕ)* *(#(F, ϕ))* for any propositional formula ϕ as follows:

#(**V**, p)	=	1	#(**F**, p)	=	1
#(**V**, ¬ϕ)	=	#(**F**, ϕ)	#(**F**, ¬ϕ)	=	#(**V**, ϕ)
#(**V**, ϕ∨ψ)	=	#(**V**, ϕ)+#(**V**, ψ)	#(**F**, ϕ∨ψ)	=	#(**F**, ϕ)•#(**F**, ψ)
#(**V**, ϕ∧ψ)	=	#(**V**, ϕ)•#(**V**, ψ)	#(**F**, ϕ∧ψ)	=	#(**F**, ϕ)+#(**F**, ψ)

The rationale for the clauses is easy to see from our earlier explanation of strategies. This is much like counting numbers of strategies for players in any finite game tree. Of course, counting becomes more complex with full first-order evaluation games.

Games and game boards The Success Lemma compares *two semantic settings*. One is the given model ***M***, or perhaps better, its associated space of assignments *s* of individual objects *s(x)* to all relevant first-order variables *x*. This space serves as a *game board*, a setting where evaluation games but also other games can be played. Compare a chess-board. This provides possible positions – but Chess itself arises by expanding this setting with a number of typical game conventions, defining turns for players, as well as their winning positions. The latter are 'game-internal': there is nothing intrinsic to the physical distribution of chess pieces on the board which makes it a 'win' for White or Black. Varying conventions, one could play many games on the board with the same pieces: perhaps more interesting ones…

Comparing two different languages The Success Lemma compares the game and its board using expressions from different languages appropriate to them:

V has a winning strategy in ***game****(ϕ, **M**, s)* iff ***M**, s* $\models \phi$

The expression on the left can be rewritten in an appropriate game language, referring to 'forcing powers' of players – while the expression on the right-hand side is best viewed as a modal formula referring to action possibilities on the board:

game*(ϕ, **M**, s)* $\models WIN_V$ iff ***M**, s* $\models \phi$

This dual perspective can be generalized. On the left, one can talk about both players' powers for forcing any set of positions in the game. This corresponds to making

nested substitutions in modal assertions about the game board on the right. This observation will return in the general game languages of Chapters 6, 7, and 8.

Defining the games formally Writing out a complete formal definition of extensive game trees for logic games is often just a boring matter of implementation. But in the present case, some formalization brings out some interesting twists. One can define the tree for ***game***$(\phi, \mathbf{M}, s)$ more precisely as follows. Nodes are all pairs

(s, ψ) where s is an $\mathbf{M}$-assignment, and ψ is a subformula of ϕ

Game *moves* reflect the earlier ones, changing one or both components of the state. In particular, atomic tests do not change the assignment component, and neither do choices. E.g., a choice move attaches a node $(s, \phi \vee \psi)$ to two daughters (s, ϕ) and (s, ψ). Formalizing the other game rules involves more interesting issues, and leads to more radical departures from received views in predicate-logical semantics.

First, consider assignment change with quantifiers. In a move starting from $(s, \exists x \psi)$, Verifier chose some object d from the domain of $\mathbf{M}$, and the current assignment s is set to $s[x:=d]$. After that, play continues with respect to $\psi(d)$: i.e., it starts afresh from the formal game state $(s[x:=d], \psi)$. But this analysis suggests that, unlike in standard logical syntax, we can view the quantifier symbol $\exists x$ itself as a separate interpretable entity, and more specifically, that

quantifiers are atomic games of object picking!

This is unlike standard logical thinking, which makes first-order quantifiers akin to Boolean disjunctions and conjunctions. The real game operation involved in evaluating formulas $\exists x \psi$ is *sequential composition*, glueing the game for the subsequent formula ψ after the independent atomic game for $\exists x$. On this view,

> Predicate-logical semantics is really a system of games of object picking and fact testing, related by suitable game operations.

Next, we need game-internal predicates of *turn taking* and *winning*. In principle, each formula ϕ tells us who is to move at which stage, but we cannot read this off directly, because of role switches with negations. This can be circumvented by pushing negations inside to the atoms. A more general solution is as follows.

Definition 1.5 Formal game trees

We define ***game***$(\phi, \boldsymbol{M}, s)$ inductively, for any assignment s – starting from an initial state (s, ϕ). The first two clauses are for the two kinds of atomic game:

(a) ***game***$(\phi, \boldsymbol{M}, s)$ for atomic ϕ is a one-node end game, which is a win for Verifier if $\boldsymbol{M}, s \models \phi$, and for Falsifier otherwise.

(b) ***game*** $(\exists x\phi, \boldsymbol{M}, s)$ is a one-move games starting at s where it is ***V***'s turn, with possible moves to any state $s[x:=d]$, always ending in a win for ***V***.

Next we turn to game constructions:

(c) ***game*** $(\phi \vee \psi, \boldsymbol{M}, s)$ is the disjoint union of the two games ***game*** $(\phi, \boldsymbol{M}, s)$ and ***game*** $(\psi, \boldsymbol{M}, s)$ put under a common root $(s, \phi \vee \psi)$ which is ***V***'s turn.

(c) ***game*** $(\phi \wedge \psi, \boldsymbol{M}, s)$ is defined likewise, but with an initial turn for ***F***

(e) ***game*** $(\boldsymbol{M}, \neg\phi, s)$ is ***game*** $(\boldsymbol{M}, \phi, s)$ with turn and win markings reversed.

The workings of the negation switch were illustrated in the proof of Lemma 1.3. Next, in addition, as shown by our analysis of quantifier games, we can now add a general clause for a useful operation of *composition* for evaluation games:

(f) ***game*** $(\phi \,;\, \psi, \boldsymbol{M}, s)$ is the tree arising as follows. First take ***game*** $(\phi, \boldsymbol{M}, s)$ and continue at end states with assignment t with a copy of ***game*** $(\psi, \boldsymbol{M}, t)$.

Clause (f) allows continuations of end positions in the first game which are not wins for Verifier. This is in order to interpret a game like $\forall x\, Px$ (or, in our new syntax $\forall x \,;\, Px$), where Falsifier has the winning positions after the first game $\forall$x.

Emancipation of syntax Our next point is a different look at first-order syntax. The above game semantics interprets much more than just the usual 'well-formed formulas'. For instance, it makes perfect sense to play a game for the string

$$Px \,;\, \exists x$$

"First test that P holds for object $s(x)$ under the current assignment s, then change the value of x, and stop". This game would be different from that for $\exists x \,;\, Px$: which first changes the value of $s(x)$ so as to obtain an object which has the property P. More generally, with composition as a new formula construction, predicate logic extends to a language of a sort of 'discourse chunks' $\alpha; \beta; \ldots$ and we have a nice

Open question Axiomatize first-order logic with an added game composition.

The new view also allows us to see new distinctions in existing first-order syntax. Consider a typical *bounded quantifier*, typical for modal logics, of the form

'for some R-successor, ϕ holds'.

Standard predicate logic writes this as $\exists y(Rxy \& \phi)$, which defines the following game. Verifier chooses an object d for y, then Falsifier chooses a conjunct, and one either tests the atom $R^{M}s(x)d$, or play continues at the game state $(s[y:=d], \phi)$. But it seems much more natural to package this as *one game move*:

Verifier must pick an R-successor of $s(x)$.

This atomic game is won by Verifier if she can produce such an object, and lost otherwise. In extended first-order syntax, this would be the game for

$\exists y \, ; \, Rxy$

One might call this broader realm of semantic interpretability 'emancipation of syntax': even the not so well-formed formulas now get their chance.

Propositions versus activities Finally, the reader should beware of a confusion. Reading logical formulas ϕ in their ordinary notation as evaluation games makes them serve a double role: once as a static *proposition* in ordinary truth-conditional semantics, and once as a *game* in our new semantics. And a game is not a statement, but a dynamic activity. To be sure, one can state propositions *about games*, say, that Verifier has a winning strategy in the ϕ-game. But this statement is not that game itself – and it does not exhaust the latter's content in any obvious sense. Much of the literature on game semantics switches carelessly between these two readings of first-order notation, resulting in the widespread prejudice that mere existence statements about Verifier's winning strategies are all there is to know to the game-theoretic meaning of ϕ.

These two levels are known from dynamic logic, which has both *programs* and *propositions* intertwined. Propositions occur inside programs, like test conditions for a loop – or they may hold about programs, like correctness assertions about execution. We will see a similar interplay in the dynamic game logic of Chapter 8.

1.4 Determinacy, finite and infinite games

Now, let us broaden the connection of evaluation games with game theory. First, through the Success Lemma, logical laws may now acquire game-theoretic import.

Finite depth games Recall our first encounter between logic and game theory from our Introduction. As was explained in Section I.3.1:

Fact 1.6 Excluded Middle $A \vee \neg A$ expresses *determinacy* of evaluation games:
that is, one of the two players must always have a winning strategy.

This is just the tip of an iceberg. Evaluation games turned out to be determined because of a few very general features, expressed in Zermelo's early result that

Theorem 1.7 Zermelo's Theorem
All zero-sum two-player games of *finite depth* are determined.

Infinite games Much stronger results than Zermelo's have been found since. These shift attention to *infinite games*, where players produce infinite runs, marked for being winning or losing. Some logic games are of this sort, as we shall see later in this chapter, and the next. But infinite games are also of independent interest. They model situations with ongoing behaviour, like some logical practice – which is just as important as finite termination. A good analogy comes from computer science. *Terminating programs* are meant to find some value, or finish some task within a finite time. But there are also crucial programs like *operating systems* whose task it is to ensure proper functioning of some device: the longer the better. Or take linguistics. There are language games evaluating specific assertions, which should terminate. But there is also the Great Game of Language: with discourse as the 'operating system of cognition', which should keep functioning forever – and where 'losing' would be a break-down in that history, or failure to allow a proper say to all participants. Thus, finite and infinite games can live happily together.

Winning conditions for runs (i.e., complete branches, finite or infinite) in infinite games may be more or less complex, depending on their description. An important special case occurs when winning a run for some player depends only on what happens in some finite initial segment of play. Here is the crucial notion.

Definition 1.8 Open sets of runs
A set of runs O is *open* if every infinite sequence that belongs to O has a finite initial segment X such that all runs of the game sharing X are also in O.

Here is a key result from the 1950s generalizing Zermelo's Theorem:

Theorem 1.9 Gale-Stewart Theorem

Games in which the set of all winning runs
for one of the players is open are determined.

Proof First we state the following completely general auxiliary observation.

Fact 1.10 Weak Determinacy

If player ***E*** has no winning strategy 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 winning strategy for the remaining game from then on.

Proof of the Fact ***A***'s strategy arises by the same reasoning as for Zermelo's Theorem. 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. ■

Now, assume without loss of generality that ***E*** is the player with the open set of winning runs. Then ***A***'s 'nuisance strategy' just described is winning for him. For consider any run *r* produced. If *r* were winning for ***E***, some finite initial segment *r(n)* would have all its continuations winning, by openness. But then, ***E*** *would* have had a winning strategy by stage *r(n):* namely, 'play whatever'. Quod non. ■

Open winning conditions are found., e.g., with logic games of model comparison and model construction. The Gale-Stewart Theorem is a special case of a much stronger fact called Martin's Theorem stating that all Borel games are determined. Here winning conditions lie in the so-called 'Borel Hierarchy' of sets of sequences. Natural examples involving non-open Borel conditions are:

(a) the operating system must be *fair*: all requests eventually get answered
(b) the parity winning condition for *fixed-point games* later in this chapter.

Non-determinacy With non-Borel winning conventions, infinite games can be *non-determined*. To define an example we need a key set-theoretic notion.

Definition 1.11 Ultrafilters

An *ultrafilter* on the natural numbers *N* (the only case we need) is a family ***U*** of non-empty sets of natural numbers with the following three properties:

(a) the family $\boldsymbol{U}$ is closed under taking supersets

(b) the family $\boldsymbol{U}$ is closed under taking intersections

(c) $X \in \boldsymbol{U}$ iff $N–X \notin \boldsymbol{U}$ for all sets $X \subseteq N$

A *free* ultrafilter is a family $\boldsymbol{U}$ of this sort which contains no finite sets.

Free ultrafilters exist by the Axiom of Choice. Let $\boldsymbol{U}^*$ be one. Here is the game:

Example 1.12 A non-determined game
Two players pick successive neighbouring closed initial segments of the natural numbers, of arbitrary finite sizes, producing a succession like this:

$\boldsymbol{A}$: $[0, n_1]$, with $n_1 > 0$, $\boldsymbol{E}$: $[n_1+1, n_2]$, with $n_2 > n_1+1$, etc.

We stipulate that player $\boldsymbol{E}$ wins if the union of all intervals chosen by her is in $\boldsymbol{U}^*$ – otherwise, $\boldsymbol{A}$ wins. Winning sets in this game are not open for either player, as sets in $\boldsymbol{U}^*$ are not determined through finite initial segments.

Fact 1.13 The interval selection game is not determined.

Proof Player $\boldsymbol{A}$ has no winning strategy. If he had one, then $\boldsymbol{E}$ could use that with a delay of one step to *copy* $\boldsymbol{A}$'s responses to her own moves disguised as $\boldsymbol{E}$'s moves. Both resulting sets of intervals (disjoint up to some finite initial segment!) would then have their unions in $\boldsymbol{U}^*$: which cannot be, as our ultrafilter was free. But neither does $\boldsymbol{E}$ have a winning strategy, for similar reasons. ■

Through the copying, modulo initial segments, players' powers are even the same in this game. We will analyze this symmetric situation a bit further in Chapters 7, 10, when studying players' powers in possibly non-determined infinite games.

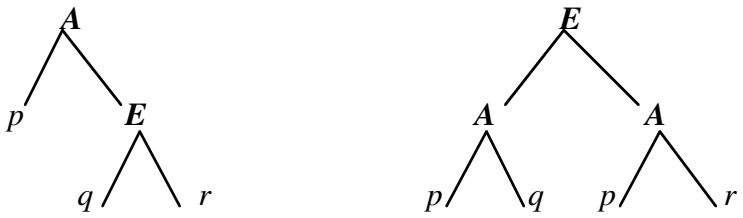
The dependence of the counter-example on the Axiom of Choice is not a pedantic observation. Set theorists have played since the 1960s with an alternative *Axiom of Determinacy*, stating instead that all infinite games are determined! There is a highly sophisticated mathematical literature on these issues in descriptive set theory (cf. Moschovakis 1980, Vervoort 2000), but its main thrust is somewhat tangential to our business with games in these lecture notes.

1.5 Game-theoretic content of predicate logic: some equivalences

Determinacy is just one of many concerns. In the Introduction, we already saw informally that valid laws of predicate logic may encode several different features of games. In particular, in Section I.3.2, we noted an issue of game equivalence.

Fact 1.14 The propositional Distribution law states that the evaluation games for its formulas are outcome-equivalent in terms of players' forcing powers.

More concretely, switching between games $A \wedge (B \vee C)$ and $(A \wedge B) \vee (A \wedge C)$ may be viewed as a transformation allowing us to interchange the order of turns for players in a game without affecting their strategic powers concerning outcomes:



More general correspondences between logical laws and game-theoretical principles in the algebra of 'power equivalence' will be studied in Chapters 7, 8. In particular, the Boolean operations then become instances of general operations on games.

Remark Implication and equivalence: inside and outside games

Should not the right notion of game equivalence be automatic once we define an evaluation game for formulas $\phi \leftrightarrow \psi$, just as for the other Boolean operations? And likewise, could we define a game for formulas $\phi \rightarrow \psi$ to get a *game implication*? This is not obvious. First, implication and equivalence are relations between games, not operations on them. Now, classical logic has the Deduction Theorem, stating that formula ϕ implies formula ψ iff the implication $\phi \rightarrow \psi$ is valid. This might suggest that two games *G, H* are equivalent iff some distinguished player has the power to win their 'equivalence game' $G \Leftrightarrow H$, for some operation $\Leftrightarrow$ yet to be defined. But which operation? Boolean equivalence will not do, as it enforces a mere top-level initial choice. One needs to relate more of the internal structure of the games *G, H*. For this purpose, the 'interleaving' *linear* implication and equivalence of Chapter 9 may be better candidates. Or, one can look at structural comparison relations like *bisimulation* in Chapter 2. We will return to this issue in Chapter 5.

Leaving these deeper issues aside, in the remainder of this section, we just discuss the general game-theoretic background for our specific Fact 1.14. Distribution is one of the key Boolean equivalences which allow us to turn any proposition into a *conjunctive normal form* $\&_i \vee_j (\neg) p_{ij}$. Now all these laws are game-valid – and so the usual logical normal form procedure applies to games as well:

Fact 1.15 Normal forms for games

The standard propositional transformation into conjunctive normal form provides an outcome-equivalent 2-move version for any finite game.

This result will be proved in an abstract version in Chapter 7. The corresponding game trees start with choices for ***A***, each followed by choices for ***E***, or vice versa. This is close to strategic game form– but there, the two moves are independent.

Other special formats have a game-theoretic sense, too. E.g., first-order *prenex forms* like $\forall x \exists y \forall z\, Q(x, y, z)$, with all quantifiers moved in front, divide all object picking moves into alternating blocks for each player, giving an alternating scheduling without affecting players' powers. Finally, consider so-called

Skolem forms, replacing first-order formulas by equivalents of the form $\exists f_1 \ldots \exists f_k \forall x_1 \ldots \forall x_m$ followed by a quantifier-free propositional part

E.g., the first-order formula $\forall x \exists y \forall z \exists u\, Q(x, y\, z\, u)$ has the Skolem form

$\exists f \exists g \forall x \forall z\, Q(x, f(x), z, g(x, z))$

In terms of the caveat of Section 1.3, this would be a typical assertion *about* the evaluation game for $\forall x \exists y \forall z \exists u\, Q(x, y, z\, u)$. It claims that ***V*** has a strategy of object picking – given by the two functions f, g – countering any moves made by ***F*** to produce atoms where ***V*** wins. Because of this, some authors *identify* winning strategies in first-order evaluation games with finite sequences of Skolem functions. We disagree: strategies are the more general notion. E.g., Skolem functions do not encode Verifier's propositional choices, needed to win a game like $\forall x\, (Px \vee Qx)$.

Remark Skolem forms as strategic games

Having said all this, perhaps somewhat perversely, one *can* view the above second-order statement $\exists f \exists g \forall x \forall z\, Q(x, f(x), y, g(x, y))$ itself as a game! This would be a second-order game over the original one, where Verifier selects a strategy right at the start, after which Falsifier is left to do the best he can by picking objects – and after that, an ordinary propositional end game is played. This, too, is an abstract transformation, going from extensive games to strategy selection games. Such redescriptions occur occasionally in logic games: we will see another in Chapter 2.

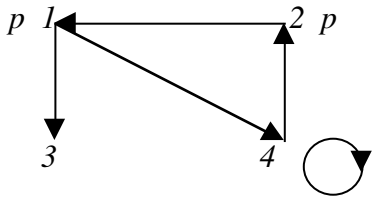
1.6 'Gamification': variations and extensions

Evaluation games are not special to first-order logic. The ideas of this chapter apply equally well to other logical languages. In fact, one can use the above explanations to provide more or less automatic 'gamifications' of the semantics for many logical languages, provided that the truth conditions employ quantifiers and connectives. E.g., the above 'Skolem game' is really just an evaluation game for a *second-order langauge*, which goes like before, except for letting players now also choose *sets* for second-order quantifiers $\exists X$, in addition to objects for first-order quantifiers $\exists x$. We will discuss two examples here with some additional point to be made. The first of these concerns, essentially, a fragment of the first-order language.

Modal logic Consider the basic *modal language* interpreted over possible worlds models $\mathbf{M} = (W, R, V)$, according to its standard truth definition:

$\mathbf{M}, s \models \langle\rangle\phi$ iff for some t with Rst, $\mathbf{M}, t \models \phi$
$\mathbf{M}, s \models [\,]\phi$ iff for all t with Rst, $\mathbf{M}, t \models \phi$

It helps to think of the worlds as 'states' of some kind, while accessibility encodes moves that can be made to get from one state to another. Here is an example:



modal formula $\langle\rangle[]\langle\rangle p$ is true in states *1, 4*, but it is false in *2, 3*

Modal evaluation games search through such a model, with these key moves:

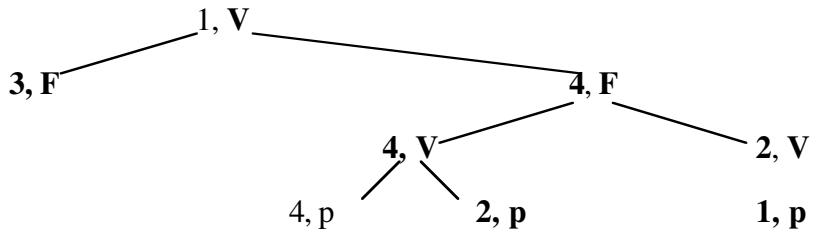
$\langle\rangle$ Verifier chooses a successor of the current world
$[\,]$ Falsifier chooses a successor of the current world

States of the game are pairs *(state, formula).* Players lose, when a proposition letter defended by them fails at the current state, or when they must choose a successor, but cannot. The analogy to keep in mind here is that between a modality $\langle\rangle p$ and a *bounded* first-order quantifier $\exists y(Rxy \,\&\, Py)$, discussed in Section 1.3.

Again, by the same proof as before, we have the obvious equivalence:

Fact 1.16 Modal Success Lemma
$\mathbf{M}, s \models \phi$ iff $\mathbf{V}$ has a winning strategy for the ϕ-game in $\mathbf{M}$ from s.

E.g., the above graph model induces the following game tree for formula $\langle\rangle[]\langle\rangle p$ starting from state *1*, with bold-face indicating winning game nodes for Verifier:



In this game, ***V*** has two winning strategies: "left", and "right", <"right", "down">. These are indeed the two ways of verifying $\langle\rangle[]\langle\rangle p$ in the given model at world *1*. Even so, there is a special feature to these modal games not found before. Consider ***F***'s plight at the bottom left. This time, players can have turns at end nodes – losing because they must move, but cannot. This is a well-known general convention.

Exactly the same stipulations work for the more common poly-modal languages with indexed operators $\langle a\rangle$, $\langle b\rangle$, … referring to alternative relations R_a, R_b, …

Fixed-point logics: modal μ-calculus The following rather technical topic concerns an extension of the modal language that will play a role in Chapters 6, 14, when analyzing the logical structure of Zermelo's Theorem, and game-theoretic equilibria. It will probably mean most to readers with some background in logics with *fixed-point operators,* defining smallest or greatest fixed-points of monotone operators corresponding to formulas. The simplest case is as follows.

Definition 1.17 Modal μ-Calculus

The modal *μ–calculus* extends the basic modal logic with an operator

$\mu p\bullet \phi(p)$ in which all occurrences of p in ϕ must be *positive*.

Positive occurrence of a proposition letter means that, counting from the outside of the formula, it lies in the scope of an even number of negations. In any model ***M***, the new operator $\mu p\bullet \phi(p)$ defines the *smallest fixed-point* w.r.t. set inclusion of the following set operator ϕ^{M} on the model associated with the formula $\phi(p)$:

$$\phi^{M}(X) = \{s\in \mathbf{M} \mid (\mathbf{M}, p:=X), s \models \phi\}$$

Here a *fixed-point* of a function F is an argument X such that $F(X)=X$.
Some modal formulas of this extended sort are

$\mu p\bullet q\vee \langle a\rangle p$, $\mu p\bullet [a]p$, $\neg\mu p\bullet \neg\langle a\rangle\neg p$

We will get to their interpretation in a moment. Stirling 1999 is a nice introduction to this language, showing how it expresses key properties of computational processes. Here is some elementary explanation. First, thanks to the positive occurrence of p in $\phi(p)$, the operator ϕ^{M} is monotonic with respect to inclusion:

$$X \subseteq Y \;\Rightarrow \phi^{M}(X) \subseteq \phi^{M}(Y)$$

This observation allows us to use some general mathematical background

Lemma 1.18 (Tarski–Knaster)

Any monotonic operator F on the subsets of a set A has both
a smallest and a greatest fixed-point in the inclusion ordering.

The proof consists in checking that

the smallest fixed-point F_* is $\quad \cap\{X \subseteq A \mid F(X) \subseteq X\}$

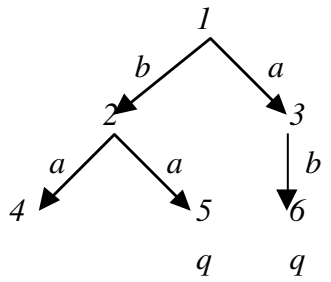
the greatest fixed-point F^* is $\quad \cup\{X \subseteq A \mid X \subseteq F(X)\}$

Alternatively, we can think of these as reached by approximation through the ordinals. E.g., is the first set where F reaches a fixed-point in the sequence

$\varnothing, F(\varnothing), FF(\varnothing), \ldots \quad F^{\omega}(\varnothing), FF^{\omega}(\varnothing), \ldots$

taking the union of all previous stages at limit ordinals

This allows us to define certain properties of states in a model by recursion. E.g., consider the first formula $\mu p \bullet q \vee \langle a \rangle p$ shown above in the following model:



Here is the approximation sequence for the associated set function of $q \vee \langle\rangle p$:

stage	set	defining formula
0	$\varnothing$	$\bot$
1	*{5, 6}*	q
2	*{5, 6, 2}*	$q \vee \langle a \rangle q$
3	*{5, 6, 2}*	the fixed-point!

In this specific model then, the fixed-point also has a standard modal definition, but the μ–formula gives greater generality. What it describes in general is the set of all point which can reach a q–world through some finite sequence of a–steps.

Remark Defining Propositional Dynamic Logic

The preceding observation shows how the μ–calculus can define the characteristic expressions of another modal system to be used in these notes, viz. *propositional dynamic logic* which allows arbitrary program expressions π inside its operators (cf. Chapters 6, 8, Harel, Kozen & Tiuryn 2000). E.g., an iteration modality $\langle a^* \rangle \phi$ says that ϕ holds at a state reachable from now after some finite number (zero or more) of a–steps. This assertion satisfies the following fixed-point equation

$$\langle a^* \rangle \phi \leftrightarrow (\phi \vee \langle a \rangle \langle a^* \rangle \phi)$$

and indeed, it may be simply defined as $\mu p \bullet \phi \vee \langle a \rangle p$.

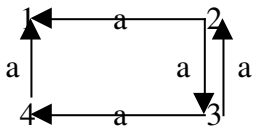
By a similar analysis, the formula $\mu p \bullet [a]p$ holds in those points in a model with at most finite sequences of a-steps coming out of them – the *well-founded part* of the binary relation R_a. One difference between the two cases is that the approximation process for the first formula will always stop in ω steps, while that for the second can go on to any ordinal, depending on the size of the model. Our third example is more complicated. It amounts to a *greatest fixed-point operator* w.r.t ϕ^{M} :

$\nu p \bullet \phi(p)$ in which all occurrences of p in ϕ must be *positive*.

E.g., one can check that, in any model, $\nu p \bullet \langle a \rangle p$ defines the set of all those points s which have an endless sequence of a-steps coming out of them: $s\ R_a\ s_1\ R_a\ s_2\ R_a \ldots$ Using general properties of fixed-points, we can show that

$\nu p \bullet \phi(p)$ is equivalent to $\neg \mu p \bullet \neg \phi(\neg p)$

This then explains the meaning of our third formula. The reader might verify that its approximation sequence will stabilize at the set *{2, 3}* in the following model:



Thus, fixed-point formulas can be non-trivial to interpret. Our syntax even allows iterated fixed-point formulas like $\mu p \bullet \mu q \bullet \phi(p, q)$, which need still more decoding.

We will not define evaluation games for the μ—calculus (cf. again Stirling 1999 how these will work), leaving this issue for our next formal language.

First-order fixed-point logic The same ideas as in the modal case work for an extension of first-order logic with fixed-point operators (cf. Moschovakis 1974, Ebbinghaus & Flum 1995). Again we have smallest fixed point operators

$\mu P \bullet \phi(P)$ with P occurring only positively in ϕ

More precisely, with $\boldsymbol{x}$ ($\boldsymbol{d}$) standing for finite tuples of variables (objects),

$[\mu P, \boldsymbol{x} \bullet \phi(P)](\boldsymbol{d})$ says that $\boldsymbol{d}$ is in the smallest set X with $\phi^{M}(X) = X$
$[\nu P, \boldsymbol{x} \bullet \phi(P)](\boldsymbol{d})$ says that $\boldsymbol{d}$ is in the largest set X with $\phi^{M}(X) = X$

For instance, here is a definition for the *transitive closure* of a relation R:

$\mu P, xy \bullet \ (Rxy \vee \exists z\,(Rxz \wedge Pzy))$

This shows in the typical fixed-point behaviour for this notion:

$\boldsymbol{trans}(R)(x, y) \leftrightarrow R(x, y) \vee \exists z\,(R(x, z) \wedge \boldsymbol{trans}(R)(z, y))$

This recursive unwinding suggests how evaluation games for this language may be interestingly different from those for first-order formulas defined in this chapter. Fixed-point games may become *infinite*. If $\boldsymbol{V}$ chooses the right-hand conjunct, and then moves to an R–successor, the original formula returns, and play loops.

Definition 1.19 First-order fixed-point games
In games for a first-order fixed-point language, Verifier and Falsifier play according to our earlier rules when the outermost operator is a first-order one. When a fixed-point formula $[\mu P, \boldsymbol{x} \bullet \phi(P)](\boldsymbol{d})$ is encountered the formula at the next stage becomes $\phi(P)(\boldsymbol{d})$ – with the understanding that, whenever further play hits an atom $P\mathbf{e}$, no evaluation takes place (P is a 'bound predicate variable'), but $[\mu P, \boldsymbol{x} \bullet \phi(P)](\boldsymbol{e})$ is substituted back in. The treatment of greatest fixed-point formulas $[\nu P, \boldsymbol{x} \bullet \phi(P)](\boldsymbol{d})$ is analogous. The technical winning convention for these games exploits the fact that, in infinite runs, certain fixed-point subformulas of the finite initial formula ϕ must have been called infinitely often. Indeed, there must be a unique 'recurrent' subformula occurring in highest syntactic position in ϕ. Then we say that

An infinite run is a win for ***V*** in the ϕ–game if the syntactically highest recurrent fixed-point subformula is a greatest fixed-point. Otherwise, if that highest recurring subformula is a smallest fixed-point, ***F*** wins.

E.g., consider the smallest fixed-point definition for transitive closure of a relation. To show that *Pde* for objects *d, e*, Verifier need not show directly that *Rde*, but may choose the second disjunct, take some object *f* for *z*, and now claim that *Pfe*. But she may only do that finitely often – otherwise, she has not made her point, and loses. The case of greatest fixed-point definitions is dual: Falsifier must 'put up' in some finite number of cycles, while it is OK for Verifier to keep the cycle going.

Proposition 1.20 Success Lemma

A formula in first-order fixed-point logic is true if and only if Verifier has a winning strategy in the game just described.

See Doets 1999 for details of the argument, which is no longer a simple induction. In the background is an earlier result. These games do not have an open winning condition for players, as the recurrence in 1.18 cannot be determined on the basis of some finite segment. So the Gale-Stewart Theorem does not apply. But the winning condition is Borel, which still makes them determined by Martin's Theorem.

Excursion 1.21 Second-order reformulation

A *finite* version of this game would translate formulas $\mu P, \boldsymbol{x}\bullet\ \phi(\boldsymbol{d})$ inside out by equivalent second-order statements. The defining condition for the smallest fixed-point in Lemma 1.18 says that ***d*** satisfies every predicate which is a smallest pre-fixed point of the monotone set operation defined by the positive formula ϕ: This can be written as the following second-order formula:

$$\forall Q\,(\,\forall \boldsymbol{x}\,(\phi(Q, \boldsymbol{x}) \rightarrow Q\boldsymbol{x}) \rightarrow\ Q\boldsymbol{d})$$

The standard evaluation game for these second-order formulas is of finite depth, though at the cost of letting players choose *sets* of tuples occasionally. (But, what do we *mean* exactly when we say this new game is 'equivalent' to the original?)

Further variations Even for the original first-order language itself, genuine variations are possible on the standard evaluation game, making players perform different tasks. E.g., in contrast to object-picking moves, one may consider moves that *withdraw objects* from a domain *without replacement* – or even moves that drastically *change a domain* or an interpretation function, by adding or subtracting

individuals and facts. After all, some real games drastically transform their playing field... Such variations mix the process of logical evaluation, leaving a model undisturbed, with model construction, which will be considered in Chapter 3.

I.7 Conclusion

This concludes our introduction to two-player evaluation games as a general dynamic perspective on logical semantics. Most of these games have finite depth, but some are naturally infinite. In addition, we have found a number of general game-theoretic themes that will return in later chapters. These include (a) determinacy, (b) game equivalence, (c) the logic of operations on games such as choice, switch, or composition, and (d) the systematic importance of strategies. In particular, reasoning about properties of evaluation games presupposes an implicit *calculus of strategies*, which we will take up in the integrative Chapter 5.

References

J. Barwise & J. Etchemendy, 1999, *Language, Proof, and Logic*, Seven Bridges, New York.

H. C. Doets, 1999, 'Evaluation Games for First-Order Fixed-Point Logic', manuscript, Institute for Logic, Language and Computation, University of Amsterdam.

H-D Ebbinghaus & J. Flum, 1995, *Finite Model Theory*, Springer, Berlin.

D. Harel, D. Kozen & J. Tiuryn, 2000, *Dynamic Logic*, The MIT Press, Cambridge (Mass.).

J. Hintikka, 1973, *Logic, Language Game,s and Information*, Clarendon Press, Oxford.

J. Hintikka & G. Sandu, 1997, 'Game-Theoretical Semantics', in J. van Benthem & A. ter Meulen, eds., *Handbook of Logic and Language*, Elsevier, Amsterdam, 361–410.

Y. Moschovakis, 1980, *Descriptive Set Theory*, North-Holland, Amsterdam.

C. Stirling, 1999, 'Bisimulation, Modal Logic, and Model Checking Games', *Logic Journal of the IGPL* 7:1, 103–124.

M. Vervoort, 2000, *Games, Walks, and Grammars*, dissertation DS 2000-03, Institute for Logic, Language and Computation, University of Amsterdam.