19
Dynamic Logic of Sequential Game Operations

This chapter is about a system that describes powers of players in complex games on the analogy of propositional dynamic logic for programs, by substituting the forcing relations of Chapter 11 for the transition relations between states. This 'dynamic game logic' was proposed in Parikh (1985). There is much of interest to how this approach to logic and games works, and we tell the story slowly, with motivations and follow-up. Much of what follows is our own take, adding results on game algebra, bisimulation, and other logical themes.

19.1 Internal and external views of games

Logical description of games can proceed at various levels. The game logics of Parts I, II, and III were game-internal, providing statements that may hold at stages of play. But the logic games of Part IV, and especially the evaluation games studied in Chapter 14, took a game-external point of view, with formulas standing for whole games, and logical operations becoming game-forming operations of choice, composition, or role switch. The external viewpoint has its attractions, especially in describing game equivalences in an algebraic manner.[214] Internal and external viewpoints can be combined, as we will see in this part of our book. To do so, we step up the abstraction level for studying games. This chapter presents a dynamic logic for external game combination, at the level of forcing and outcomes. It encodes

214 Similar options occur with computation. Modal logics provide process-internal views, while process algebra (Bergstra et al. 2001) studies global process combination. The two views are related by bisimulation techniques, but they proceed differently.

an algebra of basic game-forming operations, while providing a new scope for modal techniques such as bisimulation.

19.2 From logic games to game logics

Instead of designing logic games for specific tasks, one can also study the combinatory structure of arbitrary games. Ideas developed for logic games then acquire a more general significance. In particular, we find natural operations that structure games. For instance, choice and role switch occurred across logic games, but they are much more general than that, as basic constructions that turn given games into new ones. Next, given such operations, we want to learn their general algebraic laws. For instance, it is important to see that the propositional distributive law

$$p \wedge (q \vee r) \leftrightarrow (p \wedge q) \vee (p \wedge r)$$

is valid in complete generality. It does not hinge on playing special test games for verifier and falsifier about atomic propositions at the end, or on claims made by proponent or opponent in a dialogue. The expressions p, q, r in this equivalence might stand for any games, of arbitrary complexity, plugged in at the final stages.

As we noted in Chapter 14, in the literature on logic games, logical formulas often do double duty: as static propositions, and as games (i.e., dynamic activities). These roles must now be pulled apart, following our distinction between games themselves and assertions one can make about their behavior. The system in this chapter is inspired by the dynamic logic of programs, a system that has occurred in this book ever since Chapter 1, and the evaluation games of Chapter 14 are a good analogy to keep in mind.[215]

19.3 Forcing models, games, and game boards

We start with a semiformal tour through the style of game analysis to be presented in this chapter, touching on all the major themes that arise.

---

215 The alternative system to be presented in Chapter 20 extends linear logic, with a link to the dialogue games of Chapter 17. Together, Chapters 19 and 20 span a space merging ideas from logic games and game logics.

Dynamic game logic: A quick preview We now consider a language like that of propositional dynamic logic, PDL, in Chapter 1, but this time with expressions for two-player games (with players $\boldsymbol{E}$ and $\boldsymbol{A}$) plus propositions.

DEFINITION 19.1 Language of dynamic game logic
The language of dynamic game logic is defined by these inductive syntax rules:

Formulas F	$p \mid \neg F \mid F \vee F \mid \{G\}F$
Game expressions G	$g \mid G \cup G \mid G^d \mid G\,;\, G \mid ?F$

A point of notation In this chapter, we will use variables G with primes when thinking primarily of games, but also variables $x, y, ...$ when thinking of the pure algebraic form of laws about games. Players will be denoted by $\boldsymbol{A}$ and $\boldsymbol{E}$, thought of generically as an opposite pair. ■

The game operations in this language are choice for player $\boldsymbol{E}$, dual ^{d} for role switch, and sequential composition ;. It is useful to also define a game conjunction $G \cap G'$ of choice for player $\boldsymbol{A}$ as $(G^d \cup G'^d)^d$. Parikh's system also has a game iteration G^* like the Kleene star of PDL that we will discuss separately later. Note the recursion in the syntax. Tests $?\varphi$ turn formulas φ into game expressions, while in the opposite direction, modalities $\{\ \}$ take game expressions to formulas.

The intuitive idea of the modality is as in Chapter 11, based on forcing relations (we will define the precise models a bit later on).

DEFINITION 19.2 Dynamic forcing modality
The *forcing modality* $\{G\}\varphi$ says, at any state, that player $\boldsymbol{E}$ has a strategy for playing game G guaranteeing, against any play by the opponent, a set of outcome states all of which satisfy the formula φ. ■

To make this work precisely, we need to define the following two notions:

$\rho_G\, s, X$	player $\boldsymbol{E}$ has a strategy for playing G starting in state s that is guaranteed to end up inside the set of states X
$s \models \{G\}\varphi$	for some $X : \rho_G s, X$ and $\forall x \in X : x \models \varphi$

The key relations ρ run not from states to states, as for programs, but from states to sets of states, and we recognize the forcing relations of Chapter 11. In particular, players need not be able to force unique outcomes of games, whence the set output.

DGL = neighborhood logic + game algebra Unlike modal operators $\Diamond$ and $\Box$, the modality $\{\ \}$ satisfies no distribution for disjunction or conjunction. As is

easy to see in simple games, the following formulas are invalid:

$$\{G\}(\varphi \vee \psi) \leftrightarrow \{G\}\varphi \vee \{G\}\varphi, \qquad \{G\}(\varphi \wedge \psi) \leftrightarrow \{G\}\varphi \wedge \{G\}\psi$$

What remains valid is upward monotonicity:

$$\{G\}\varphi \text{ implies } \{G\}\psi \text{ for any weaker proposition } \psi \text{ implied by } \varphi.$$

Going beyond Chapter 11, through its valid laws on top of the neighborhood logic, the logic encodes information about the basic game operations.[216] One simple example is the validity of commutativity of choice:

$$\{G \cup G'\}\varphi \leftrightarrow \{G' \cup G\}\varphi$$

Indeed, most of Boolean algebra holds for $\{\cup, ^d\}$. As for the further game operations, the forcing relations generalize standard relational algebra for relations of type $S \times S$, taking it to an algebra of relations of type $S \times P(S)$, with basic laws such as associativity:

$$\{G_1 ; (G_2 ; G_3)\}\varphi \leftrightarrow \{(G_1 ; G_2) ; G_3)\}\varphi$$

We will see much more of the mechanics of this system in Section 19.4.

Now let us become more precise about the semantic setting.

Forcing models, games, and game boards Models for this language are tuples

$$\boldsymbol{M} = (S, \{\rho_g \mid g \text{ atomic}\}, V)$$

with S a set of states, V a valuation assigning truth values to atomic propositions in states, and with atomic relations $\rho_g s, X$ assigned to basic game expressions g. These forcing relations ρ_g do not indicate players explicitly. One can take player $\boldsymbol{E}$ in mind, using determinacy of the game to find the powers of $\boldsymbol{A}$, in a way to be explained below, although we will also consider player-indexed forcing later. Forcing relations are closed under supersets, given their intended forcing interpretation:

$$\text{if } \rho_g s, X \text{ and } X \subseteq Y, \text{ then } \rho_g s, Y.$$

---

216 In what follows, in line with the original system, we start by thinking of determined games, but this restriction will soon be lifted.

In terms of the earlier discussions in Chapters 1, 11, and 18, these models $\boldsymbol{M}$ are *game boards* or playgrounds, with states that model the external content of internal game states. One board can support many games with different turns and winning positions. For instance, with evaluation games, board states are variable assignments that change when a player makes a move $x := d$. But these states do not encode the internal changes with choices between disjuncts or conjuncts.

In line with this, atomic forcing relations ρ_g do not reflect structured games g, although the latter may arise in applications of the logic. This differs from the view in Chapter 11, where forcing relations for games used internal nodes and moves.

Computing complex forcing relations To get some generality that will be useful later when dropping the assumption of determinacy, we change the above models a bit, computing forcing relations for both players in one simultaneous recursion.

DEFINITION 19.3 Forcing relations for composite games
The following recursive clauses govern forcing relations:

$$\begin{array}{lll}
\rho^{\boldsymbol{E}}_{G\cup G'}x, Y & \text{iff} & \rho^{\boldsymbol{E}}_{G}x, Y \text{ or } \rho^{\boldsymbol{E}}_{G'}x, Y \\
\rho^{\boldsymbol{E}}_{G\cup G'}x, Y & \text{iff} & \rho^{\boldsymbol{A}}_{G}x, Y \text{ and } \rho^{\boldsymbol{A}}_{G'}x, Y \\
\rho^{\boldsymbol{E}}_{G^d}x, Y & \text{iff} & \rho^{\boldsymbol{A}}_{G}x, Y \\
\rho^{\boldsymbol{A}}_{G^d}x, Y & \text{iff} & \rho^{\boldsymbol{E}}_{G}x, Y \\
\rho^{\boldsymbol{E}}_{G\,;\,G'}x, Y & \text{iff} & \exists Z : \rho^{\boldsymbol{E}}_{G}x, Z \;\&\; \forall z \in Z : \rho^{\boldsymbol{E}}_{G'}z, Y \\
\rho^{\boldsymbol{A}}_{G\,;\,G'}x, Y & \text{iff} & \exists Z : \rho^{\boldsymbol{A}}_{G}x, Z \;\&\; \forall z \in Z : \rho^{\boldsymbol{A}}_{G'}z, Y
\end{array}$$

These clauses are probably self-evident, but they will return in the soundness for the axioms of dynamic game logic in Section 19.4. ■

Henceforth, we assume that atomic forcing relations satisfy the earlier-mentioned superset closure (monotonicity) as well as consistency (cf. Chapter 12):

$$\text{if } \rho^{\boldsymbol{E}}_{G}s, Y \text{ and } \rho^{\boldsymbol{A}}_{G}s, Z, \text{ then the sets } Y \text{ and } Z \text{ overlap.}$$

One can easily show the following lifting result by induction.

FACT 19.1 If the atomic relations on a game board satisfy monotonicity and consistency, then so do all complex forcing relations as defined here.

REMARK From determinacy to nondeterminacy
Parikh (1985) assumed that all games were determined, in the sense of Chapter 11:

$$\text{for each set } Y, \text{ either } \boldsymbol{E} \text{ can force } Y, \text{ or } \boldsymbol{A} \text{ can force } S - Y.$$

Then we just define forcing for player $\boldsymbol{E}$, simplifying the case of dual to $\rho^{\boldsymbol{E}}_{G^d}x, Y$ iff not $\rho^{\boldsymbol{E}}_{G}x, S - Y$. The relations for $\boldsymbol{A}$ are induced by $\rho^{\boldsymbol{A}}_{G^d}x, Y$ iff not $\rho^{\boldsymbol{E}}_{G}x, S - Y$.[217]

Test games Finally, we must define forcing relations for atomic test games. Parikh's original stipulation was as follows (see Pauly (2001) for more explanation):

$$\rho^{\boldsymbol{E}}_{?P}x, Y \quad \text{iff} \quad x \in Y \text{ and } P \text{ holds of } y$$

This has some debatable effects, such as the equivalence $\rho^{\boldsymbol{A}}_{?P}x, Y$ iff, if $x \in Y$, then P holds of y. Hence some authors have sidestepped the issue, dropping tests altogether, and having just a special "idle game" ι with:

$$\rho^{\boldsymbol{E}}_{\iota}x, Y \;\text{ iff }\; x \in Y, \qquad\qquad \rho^{\boldsymbol{A}}_{\iota}x, Y \;\text{ iff }\; x \in Y$$

In such a game, both players have the same power, that is, the power of being there. Information about propositions P holding at the current state x is then external, and not part of the game. Another line is to have two forcing modalities. The one so far relates only to players' powers of making sure the game ends in some set of positions. Since a test does not change the current state, its correct axiom is

$$\{?P\}\varphi \leftrightarrow \varphi$$

But another view of the forcing modality $\{G, \boldsymbol{E}\}\varphi$ brings in a game-internal property: player $\boldsymbol{E}$ has a strategy for playing G guaranteeing a set of outcome states that satisfy φ and are winning for $\boldsymbol{E}$. This will validate the following two equivalences:

$$\{?P, \boldsymbol{E}\}\varphi \;\leftrightarrow\; P \wedge \varphi \qquad\qquad \{?P, \boldsymbol{A}\}\varphi \;\leftrightarrow\; \neg P \wedge \varphi$$

The second reading satisfies the same laws for game operations as the first.[218] In what follows, we merely assume that some satisfactory interpretation is in place.

Things become clearer by looking at some basic motivating examples. The above system can be used in analyzing specific games, seeing what is general, and what are peculiarities of special states and moves.

---

217 To be consistent here, one has to check that the forcing relations for compound games remain determined, when starting from determined games at the bottom level.

218 In particular, note how the winning condition reverses when we move from $\boldsymbol{E}$ to $\boldsymbol{A}$.

EXAMPLE 19.1 First-order logic
We start with the first-order evaluation games of Chapter 14 for formulas φ on models $\boldsymbol{M}$. Their internal states were pairs

$$\langle \textit{current subformula}, \textit{current variable assignment} \rangle.$$

The external board states are the variable assignments. Next, we split general formula games into atomic games plus general constructions:

$$\text{role switch } {}^{d}, \text{ two choices } \cup, \cap, \text{ plus composition } ;$$

The atomic games were of two kinds:

(a) *atomic tests* $?P$ checking whether P holds under the current assignment.

(b) *object picking* for quantifiers $\exists x$, changing the current assignment s to a new assignment $s[x := d]$ by setting x equal to some object d.

Atomic tests can be seen as games for $\{\boldsymbol{E}, \boldsymbol{A}\}$ in various ways, making sure at states where P holds that the win is for $\boldsymbol{E}$, and elsewhere for $\boldsymbol{A}$. They have special properties, not shared with games in general. For example, it does not matter in which order we perform atomic tests, and performing the same test twice does not change the outcomes:

$$?P\,;\,?Q \;=\; ?Q\,;\,?P, \qquad\qquad ?P\,;\,?P \;=\; ?P$$

Atomic quantifier games are special, too, in that player $\boldsymbol{E}$ has complete control over outcomes: $\boldsymbol{E}$ can make sure that any next state occurs. Formally, their forcing relations are "distributive," satisfying a splitting condition:

$$\text{if } \rho^{\boldsymbol{E}}_{G^d} s, \bigcup_{i \in I} Y_i, \text{ then for some } i \in I,\ \rho^{\boldsymbol{E}}_{G^d} s, Y_i$$

By contrast, for atomic quantifier games $\exists x$, the passive player $\boldsymbol{A}$ has just one power, being the total set of all x-variants of the current assignment s.

Given these stipulations for atomic games, our earlier recursive clauses compute forcing relations for any first-order evaluation game $\boldsymbol{game}(\varphi, \boldsymbol{M})$. It is also easy to see that the result squares with the success lemma of Chapter 14. Starting the game from assignment s, $\boldsymbol{E}$ has a winning strategy if and only if $\boldsymbol{M}, s \models \varphi$. ■

Logical laws deconstructed All this is not mere reanalysis of what is known. As we have seen in Chapter 14, the game view decomposes the laws of first-order

logic into several layers. Some are general game validities, having nothing to do with specific games of fact testing or object picking. This is true for our running example of Boolean distribution, or idempotence of choice: $\varphi \vee \varphi \leftrightarrow \varphi$. Other laws are special effects of the atomic repertoire, such as idempotence of composition for tests or quantifier games. Its game form $G\,;\,G = G$ fails as a general law. Still other laws may be called intermediate, such as distribution for existential quantifiers:

$$\exists x(\varphi \vee \psi) \;\leftrightarrow\; \exists x\varphi \vee \exists x\psi$$

In terms of games, its general form is

$$G\,;\,(G' \cup G'') = (G\,;\,G') \cup (G\,;\,G'')$$

This is not a valid law. We can see this quickly by substituting another game for $\boldsymbol{A}$: for example, a universal quantifier game. We then get the invalid $\forall x(\varphi \vee \psi) \leftrightarrow \forall x\varphi \vee \forall x\psi$. The special reason for the validity is the above distributive character of the game $\exists x$. Since distributivity is a ubiquitous property, its game law may be considered intermediate in force.

Thus, from a game perspective, the usual predicate-logical validities are a mixed bag. General repercussions of this observation will be found in Chapter 24.

EXAMPLE 19.2 Modal logic

Other samples of this style of analysis are modal and dynamic logic. This time, modalities are atomic games. Existential modalities $\langle a\rangle$ make player $\boldsymbol{E}$ choose some R_a-successor of the current state, universal $[a]$ are duals for player $\boldsymbol{A}$.[219] This time, however, $\boldsymbol{E}$ can lose, when there is no such successor. Thus, we have games where players may have to move, but cannot, and there is an issue of defining the proper forcing relations again. If we disregard winning conditions, we get

$$\begin{array}{lll} \rho^{\boldsymbol{E}}_{\langle a\rangle} x, Y & \text{iff} & \exists y(R_a xy \wedge y \in Y) \\ \rho^{\boldsymbol{A}}_{\langle a\rangle} x, Y & \text{iff} & R_a[x] \subseteq Y \end{array}$$

Both the first-order and the modal perspective will return in our analysis of the game algebra of sequential operations in Chapter 24.

---

219 This follows from the "standard translation" sending $\langle a\rangle q$ to $\exists y(R_a xy \wedge Qy)$.

19.4 Dynamic game logic

Basic axiom system Here is the basic proof system for dynamic game logic, DGL. It reasons about games by exploiting analogies with dynamic logic, PDL. We consider only the operations of $\boldsymbol{E}$'s choice ($\cup$), composition (;), and dual (^{d}), with $G_1 \cap G_2$ defined as above. For simplicity, we start with a version of the system for determined games.

DEFINITION 19.4 Dynamic logic for determined games
The *minimal dynamic game logic* (also called DGL) for determined games has the following principles:

(a) All valid principles of propositional logic: both axioms and rules.
(a) Monotonicity: if $\varphi \to \psi$ is provable, then so is $\{G\}\varphi \to \{G\}\psi$.
(a) Reduction laws for existence of strategies in compound games:

$$\begin{aligned} \{G\,;\,G'\}\varphi &\leftrightarrow \{G\}\{G'\}\varphi \\ \{G \cup G'\}\varphi &\leftrightarrow \{G\}\varphi \vee \{G'\}\varphi \\ \{?P\}\psi &\leftrightarrow P \wedge \psi \\ \{G^d\}\varphi &\leftrightarrow \neg\{G\}\neg\varphi \end{aligned}$$

A nondetermined version of this system will be presented below. ■

How it works Dynamic game logic encodes a good deal of "game algebra."

DEFINITION 19.5 Validity of game-algebraic identities
Two game expressions G and G' are *equal* in the sense of DGL if the following formula is valid: $\{G\}q \leftrightarrow \{G'\}q$, for some new proposition letter q. ■

This says that player $\boldsymbol{E}$ has the same powers for forcing outcomes in both games. In determined DGL, this implies that $\boldsymbol{A}$ also has the same powers.[220]

220 In nondetermined games, $\boldsymbol{E}$-equivalence by itself does not imply equivalent powers for both players: we will see a nice illustration in Chapter 21.

if $\{G\}q \leftrightarrow \{G'\}q$	is provable, then so is
$\{G\}\neg q \leftrightarrow \{G'\}\neg q$	by the substitution rule, whence
$\neg\{G\}\neg q \leftrightarrow \neg\{G'\}\neg q$	using propositional logic, and
$\{G^d\}q \leftrightarrow \{G'^d\}q$	by the axiom for dual.

To convey a sense of the mechanics of dynamic game logic, here are a few formal derivations for principles of general game algebra.

Game conjunction Intuitively, if $\boldsymbol{E}$ is to have a strategy guaranteeing φ, then one is needed in both games: $\{x\}\varphi \wedge \{y\}\varphi$. Using the definition of $\boldsymbol{A}$'s choice game $x \cap y$, we prove this:

$$\{x \cap y\}q \leftrightarrow \{(x^d \cup y^d)^d\}q \leftrightarrow \neg\{x^d \cup y^d\}\neg q \leftrightarrow \neg(\{x^d\}\neg q \vee \{y^d\}\neg q)$$
$$\leftrightarrow \neg\{x^d\}\neg q \wedge \neg\{y^d\}\neg q \leftrightarrow \neg\neg\{x\}\neg\neg q \wedge \neg\neg\{y\}\neg\neg q \leftrightarrow \{x\}q \wedge \{x\}q$$

Boolean distribution Next, consider distribution $(x \vee y) \wedge z = (x \wedge z) \vee (y \wedge z)$, that was analyzed informally earlier in this book using the definition of forcing:

$$\{(x \cup y) \cap z\}q \leftrightarrow \{x \cup y\}q \wedge \{z\}q \leftrightarrow (\{x\}q \vee \{y\}q) \wedge \{z\}q$$
$$\leftrightarrow (\{x\}q \wedge \{z\}q) \vee (\{y\}q \wedge \{z\}q) \leftrightarrow \{x \cap z\}q \vee \{y \cap z\}q$$
$$\leftrightarrow \{(x \cap z) \cup (y \cap z)\}q$$

Disjunction over conjunction (i.e., my choice over your choice) is proved similarly.

Composition and choice With distribution of sequential composition over choice, things are different. Here is a formal derivation for one version:

$$\{(x \cup y)\,;\, z\}q \leftrightarrow \{x \cup y\}\{z\}q \leftrightarrow \{x\}\{z\}q \vee \{y\}\{z\}q \leftrightarrow \{x\,;\, z\}q \vee \{y\,;\, z\}q$$
$$\leftrightarrow \{(x\,;\, z) \cup (y\,;\, z)\}q$$

And here is a failed attempt for the other version (already discarded earlier on):

$$\{x\,;\, (y \cup z)\}q \leftrightarrow \{x\}\{y \cup z\}q$$
$$\leftrightarrow \{x\}(\{y\}q \vee \{z\}q) \leftrightarrow \text{(not in DGL: only in PDL) } \{x\}\{y\}q \vee \{x\}\{z\}q$$
$$\leftrightarrow \{x\,;\, y\}q \vee \{x\,;\, z\}q \leftrightarrow \{(x\,;\, y) \cup (x\,;\, z)\}q$$[221]

221 Dropping the distribution law $\{G\}(\varphi \vee \psi) \leftrightarrow \{G\}\varphi \vee \{G\}\psi$ while retaining the other $\{G \cup G'\}\varphi \leftrightarrow \{G\}\varphi \vee \{G'\}\varphi$, is reminiscent of process algebra (cf. Milner 1989).

Dual of composition Finally, here is a valid derivation for a key law of game dual:

$$\{(x\,;\,y)^d\}q \leftrightarrow \neg\{x\,;\,y\}\neg q \leftrightarrow \neg\{x\}\{y\}\neg q \leftrightarrow \neg\{x\}\neg\neg\{y\}\neg q \leftrightarrow \neg\{x\}\neg\{y^d\}q \leftrightarrow \{x^d\}\{y^d\}q \leftrightarrow \{x^d\,;\,y^d\}q$$

Meta-theorems The key results about DGL are in Parikh (1985) and Pauly (2001).

THEOREM 19.1 Universal validity in dynamic game logic over forcing models with iteration, but without dual, is effectively axiomatized by the above DGL laws plus the iteration axioms of dynamic logic.

The relevant iteration axioms for the DGL modality will be stated below.

THEOREM 19.2 Dynamic game logic is decidable.

The proof of the first result uses a combination of standard neighborhood methods for completeness plus ideas from the completeness proof for PDL (cf. Harel et al. 2000), and we will give an impression in the final section of this chapter, using a nondetermined two-player version. Whether this also works for the full language including game dual is a longstanding open problem.

The second result follows by an effective reduction of DGL validity to PDL validity, in a way suggestive of the reasoning in Chapter 24 (cf. also Goranko 2003). The precise complexity of the SAT problem for this game logic is still unknown.

The general version Dropping determinacy gives an elegant version of DGL with separate modalities $\{G, \boldsymbol{E}\}\varphi$ and $\{G, \boldsymbol{A}\}\varphi$. Here are the basic axioms:

$$\begin{array}{lcl}
\{G \cup G', \boldsymbol{E}\}\varphi & \leftrightarrow & \{G, \boldsymbol{E}\}\varphi \vee \{G', \boldsymbol{E}\}\varphi \\
\{G \cup G', \boldsymbol{A}\}\varphi & \leftrightarrow & \{G, \boldsymbol{A}\}\varphi \wedge \{G', \boldsymbol{A}\}\varphi \\
\{G^d, \boldsymbol{E}\}\varphi & \leftrightarrow & \{G, \boldsymbol{A}\}\varphi \\
\{G^d, \boldsymbol{A}\}\varphi & \leftrightarrow & \{G, \boldsymbol{E}\}\varphi \\
\{G\,;\,G', \boldsymbol{E}\}\varphi & \leftrightarrow & \{G, \boldsymbol{E}\}\{G', \boldsymbol{E}\}\varphi \\
\{G\,;\,G', \boldsymbol{A}\}\varphi & \leftrightarrow & \{G, \boldsymbol{A}\}\{G', \boldsymbol{A}\}\varphi
\end{array}$$

Of the domestic rules for forcing relations as such, we retain

$$\begin{array}{ll}
\{G, \boldsymbol{E}\}\varphi \to \{G, \boldsymbol{E}\}(\varphi \vee \psi) & \text{upward monotonicity} \\
\{G, \boldsymbol{A}\}\varphi \to \{G, \boldsymbol{A}\}(\varphi \vee \psi) & \text{upward monotonicity} \\
\{G, \boldsymbol{E}\}\varphi \to \neg\{G, \boldsymbol{A}\}\neg\varphi & \text{consistency}
\end{array}$$

Practical uses and meta-properties of this version are similar to those above.

Iterated games DGL has a further operation of finite iteration G^*, satisfying the two standard PDL axioms of Chapter 1. In our notation, these are

$$\{G^*\}\varphi \leftrightarrow \varphi \wedge \{G\}\{G^*\}\varphi \qquad \text{fixed point axiom}$$
$$(\varphi \wedge \{G^*\}(\varphi \rightarrow \{G\}\varphi)) \rightarrow \{G^*\}\varphi \qquad \text{induction axiom}$$

The iteration game lets player $\boldsymbol{E}$ play some finite number of runs of game G where $\boldsymbol{E}$ need not say in advance how many, while infinite runs are blamed on $\boldsymbol{A}$.[222]

This concludes our presentation of the basics of dynamic game logic. Now we add some topics, linking up with earlier themes in Parts I and IV of this book.

19.5 Basic game algebra

As we have seen, the forcing semantics validates a basic algebra of game construction. Take a language of game expressions with variables for games and operations $\cup, ^d, ;$. In addition, we add a name $\boldsymbol{\iota}$ for the idle game, staying at the same state.

DEFINITION 19.6 Algebraic validity
An identity between two game expressions $G = G'$ is *valid* if interpreting these expressions in any DGL model gives both players the same forcing relations. The useful auxiliary relation $G \leq G'$ denotes the analogous valid inclusion.[223] ■

We now explore some basic validities, following van Benthem (1999).

FACT 19.2 The following principles are valid in game algebra:

(a) "De Morgan algebra" for disjunction, conjunction, and negation, whose laws are defined separately below.

---

222 Other notions of iteration occur in this book: in the evolutionary games of our Introduction, infinite evaluation games for fixed point logics in Chapter 14, or games of model comparison in Chapter 15, where it is $\boldsymbol{E}$ who wants to keep the game going forever. It is an interesting problem how to incorporate these into DGL.

223 One natural variation that does not matter to the algebraic laws would define validity by requiring forcing bisimulation, introduced below, between the games denoted by two terms in different models.

(b)	$G\,;\,(G'\,;\,G'') \;= (G\,;\,G')\,;\,G''$	associativity
	$(G \cup G')\,;\,G'' = (G\,;\,G'') \cup (G'\,;\,G'')$	left-distribution
	$(G\,;\,G')^d \quad\;\; = G^d\,;\,G'^d$	dualization
(c)	*if* $G \leq G'$, *then* $H\,;\,G \leq H\,;\,G'$	right-monotonicity
(d)	$G\,;\,\boldsymbol{\iota} \;=\; G \;=\; \boldsymbol{\iota}\,;\,G$	unit game

DEFINITION 19.7 De Morgan algebra
The system of De Morgan algebra has the standard axioms for a distributive lattice, plus an idempotent negation:

$x \cup x = x$	$x \cap x = x$
$x \cup y = y \cup x$	$x \cap y = y \cap x$
$x \cup (y \cup z) = (x \cup y) \cup z$	$x \cap (y \cap z) = (x \cap y) \cap z$
$x \cup (y \cap z) = (x \cup y) \cap (x \cup z)$	$x \cap (y \cup z) = (x \cap y) \cup (x \cap z)$
$(x^d)^d = x$	
$(x \cup y)^d = x^d \cap y^d$	$(x \cap y)^d = x^d \cup y^d$

Non-valid in De Morgan algebra are excluded middle and non-contradiction. This makes sense. For instance, as we saw in the Introduction, for games in general, $x \vee \neg x$ is no longer valid, since the game x need not be determined. ■

In this axiomatization, the crucial second set of laws is reminiscent of relational algebra (with ; behaving like composition of binary relations),[224] but the behavior of the game dual is sui generis. Further valid identities are derivable from these principles by algebraic manipulation.[225] Soundness follows by direct inspection, or via a more technical route sketched in the next remark.

REMARK Soundness via translation
A link with classical logic clarifies the above results. Starting from relation symbols $\rho_a^{\boldsymbol{E}} xY, \rho_a^{\boldsymbol{A}} xY$ for basic game expressions, write the power relations for both players in complex games using the earlier recursive clauses. These expressions are logically equivalent for both game terms in the given algebraic laws. For example, propositional distribution follows by Boolean propositional distribution applied to power relation formulas. Similarly, left distribution follows by mere predicate logic.

---

224 Game algebra lacks the right distribution: $G\,;\,(G' \cup G'') = (G\,;\,G') \cup (G\,;\,G'')$ of relation algebra. The reason for this failure has already been explained.

225 The reader may want to try the case of $(G \cap G')\,;\,G'' = (G\,;\,G'') \cap (G'\,;\,G'')$.

EXAMPLE 19.3 Non-valid principles
The translation is illustrated by two earlier-mentioned non-valid principles.

(a) $G \cap (G' \cup \neg G') = G$. On the left, one has $\rho^{\boldsymbol{E}}_G xY \wedge (\rho^{\boldsymbol{E}}_{G'} xY \vee \rho^{\boldsymbol{A}}_{G'} xY)$ for $\boldsymbol{E}$'s powers. This is not equivalent to the formula $\rho^{\boldsymbol{E}}_G xY$ on the right: $\rho^{\boldsymbol{E}}_{G'} xY \vee \rho^{\boldsymbol{A}}_{G'} xY$ is not a game tautology.

(b) $G\,;\,(G' \cup G'') = (G\,;\,G') \cup (G\,;\,G'')$. On the left, player $\boldsymbol{E}$ gets $\exists Z : \rho^{\boldsymbol{E}}_G xZ \wedge \forall z \in Z : (\rho^{\boldsymbol{E}}_{G'} zY \vee \rho^{\boldsymbol{E}}_{G''} zY)$. This is not equivalent to $\boldsymbol{E}$'s powers on the right-hand side, given by $\exists Z(\rho^{\boldsymbol{E}}_G xZ \wedge \forall z \in Z \rho^{\boldsymbol{E}}_{G'} zY) \vee \exists Z(\rho^{\boldsymbol{E}}_G xZ \wedge \forall z \in Z : \rho^{\boldsymbol{E}}_{G'} zY)$. ■

The following result shows that our analysis is on the mark.

THEOREM 19.3 Basic game algebra is complete for algebraic validity.

Proof One uses a reduction of game algebra to modal logic, reducing forcing modalities $\{\ \}$ to modal combinations $\Diamond\Box$. See Goranko (2003) for this technique, going back to Parikh (1985), that will also be used in Chapter 24.[226] ■

Open problems remain. For instance, how does the algebra change when we assume determinacy for all games? In Chapters 20 and 21, we will study laws for a richer repertoire, including parallel operations on games.

19.6 Bisimulation, invariance, and safety

In Chapters 1 and 11, we raised the issue of when two games are the same in a natural sense, finding answers in versions of bisimulation. The language of dynamic game logic does not describe games directly, but we can ask when two game boards are the same for DGL.

DEFINITION 19.8 Forcing bisimulation
A *forcing bisimulation* between two models $\boldsymbol{M}$ and $\boldsymbol{N}$ is any binary relation E between their states that satisfies the following conditions:

(a) *Atomic harmony* If xEy, then x and y verify the same proposition letters.

(b) *Back-and-forth* For each player i and each atomic forcing relation:

(i) If xEy and $\rho^{\boldsymbol{M},i}_g x, U$, then there is a V with $\rho^{\boldsymbol{N},i}_g y, V$ & $\forall v \in V \exists u \in U : uEv$.

---

226 Venema (2003) presents a more purely algebraic proof.

(ii) If xEy and $\rho_g^{\boldsymbol{N},i}y, V$, then there is a U with $\rho_g^{\boldsymbol{M},i}x, U$ & $\forall u \in U \exists v \in V : uEv$

This is essentially the notion of power bisimulation from Chapter 11. ■

Forcing bisimulation fits well in the context of this chapter.

FACT 19.3 (a) The DGL language is invariant for forcing bisimulations. (b) If two finite models $\boldsymbol{M}, s$ and $\boldsymbol{N}, t$ satisfy the same DGL formulas, then there exists a forcing bisimulation E between them with sEt.

Proof The inductive proof of (a) follows the invariance facts in Chapter 1. The proof of (b) follows the analogous fact about power bisimulation in Chapter 11. ■

As with PDL programs in Chapter 1, the proof of (a) involves a notion of safety with forcing relations for complex game terms, that we will now explain.

Safety for bisimulation In proving the invariance proposition, we need a special step. A forcing bisimulation E only guarantees the correct back-and-forth behavior for atomic relations ρ_g. But in proving that arbitrary formulas $\{G\}\varphi$ are invariant, we need back-and-forth behavior of E for all relations $\rho_G^{\boldsymbol{M}}$ for complex game expressions G in our models. The latter were constructed out of the former by our recursive rules. At this point, we introduce a notion that is known from dynamic logic (van Benthem 1996).

DEFINITION 19.9 Safety for forcing bisimulation
A game operation $G\#G'$ is *safe for forcing bisimulation* if for any forcing bisimulation E between models that satisfies the above back-and-forth conditions for the power relations $\rho_G, \rho_{G'}$, E also satisfies these conditions for $\rho_{G\#G'}^{\boldsymbol{M}}$. ■

We state the following results for forcing in determined games, but our arguments also work for the nondetermined two-player format.

FACT 19.4 All operations of DGL are safe for forcing bisimulation.

Proof Suppose that a relation E bisimulates between models $\boldsymbol{M}$ and $\boldsymbol{N}$ with respect to the two forcing relations ρ_G and $\rho_{G'}$ (\$). We show that it also bisimulates for the defined relations $\rho_{G\cup G'}$, $\rho_{G\,;G'}$, and ρ_{G^d}, and that for both players.

Case (a1) Let xEz, and $\rho_{G\cup G'}^{\boldsymbol{M},\boldsymbol{E}}x, Y$. Then either $\rho_G^{\boldsymbol{M},\boldsymbol{E}}x, Y$ or $\rho_{G'}^{\boldsymbol{M},\boldsymbol{E}}x, Y$: say, the former. By (\$) there is a set V such that $\rho_G^{\boldsymbol{N},\boldsymbol{E}}x, V$ and every point $v \in V$ has an E-related point $y \in Y$. But then a fortiori also $\rho_{G\cup G'}^{\boldsymbol{N},\boldsymbol{E}}x, V$, while every $v \in V$ still has an E-related point $y \in Y$. The other direction goes analogously. *Case (a2)* We need to show the same invariance for player $\boldsymbol{A}$. In this case, the given set Y has

both $\rho_G^{\boldsymbol{M},\boldsymbol{A}}x, Y$ and $\rho_{G'}^{\boldsymbol{M},\boldsymbol{A}}x, Y$. By (\$), we can find suitably matching sets V and W in $\boldsymbol{N}$, and the union of these is the required total match for Y, being a power for player $\boldsymbol{A}$ in both games.

Case (b) Let xEz, while $\rho_{G;G'}^{\boldsymbol{M},\boldsymbol{E}}x, Y$. By the definition of forcing relations, there is a set U with $\rho_G^{\boldsymbol{M},\boldsymbol{E}}x, U$ and $\forall u \in U : \rho_{G'}^{\boldsymbol{M},\boldsymbol{E}}u, Y$. Using (\$) once more, there is a set V with $\rho_G^{\boldsymbol{N},\boldsymbol{E}}z, V$ where every $v \in V$ has an E-related $u \in U$. Again by (\$), this time applied to the links uEv, there exist sets W_v with $\rho_{G'}^{\boldsymbol{N},\boldsymbol{E}}v, W_v$ while every $w \in W_V$ has an E-related $y \in Y$. Now, the union of all of these sets W_v is a set W that serves as a counterpart for the initial Y in $\boldsymbol{M}$, as is easy to check. The opposite direction, and also the cases for player $\boldsymbol{A}$, are similar.

Case (c) Finally, let xEz, while also $\rho_{G^d}^{\boldsymbol{M},\boldsymbol{E}}x, Y$. By the definition of determined forcing relations, $\neg\rho_G^{\boldsymbol{M},\boldsymbol{E}}x, (M-Y)$, where M is the universe of $\boldsymbol{M}$. To find a match to Y, consider the set $V = E[Y]$, where $E[Y]$ consists of all points z in $\boldsymbol{N}$ that are E-related to some $y \in Y$:

CLAIM $\rho_{G^d}^{\boldsymbol{N},\boldsymbol{E}}z, V$ and $\forall v \in V \exists y \in Y : yEv$.

The second conjunct is clear from the definition. To prove the first, recall its meaning: $\neg\rho_G^{\boldsymbol{N},\boldsymbol{E}}z, (N-V)$ with N the universe of $\boldsymbol{N}$. Suppose that $\rho_G^{\boldsymbol{N},\boldsymbol{E}}z, (N-V)$. By assumption (#), there must be a set U in $\boldsymbol{M}$ such that $\rho_G^{\boldsymbol{M},\boldsymbol{E}}x, U$ while every $u \in U$ has an E-related point $v \in (N-V)$. In particular, this means that the set U is disjoint from the initial Y. This is so because any point in U is E-related to some point v in $(N - E[Y])$, where by definition, the latter has no E-relations with Y-points. By monotonicity then, we would have that $\rho_G^{\boldsymbol{M},\boldsymbol{E}}z, (M-Y)$. But this contradicts the initial assumption. ■

Not all operations are safe for forcing bisimulation, and we give an illustration.

EXAMPLE 19.4 Unsafe game operations
A counterexample is *thirteen*(G), letting players force just those sets of states that contain at least thirteen elements. It is easy to find a concrete counterexample, using the fact that forcing bisimulations cannot count numbers of states.

Natural operations and bisimulation Our analysis illustrates a general desideratum for a process theory: the choice of an operational repertoire must fit the relevant notion of structural equivalence. For instance, van Benthem (1996) studied safety for complex programs viewed as binary state-to-state relations, finding the following theorem on expressive completeness of the PDL repertoire.

THEOREM 19.4 The first-order definable operations on programs that are safe for bisimulation are precisely those that can be defined by arbitrary applications of union, composition, test negation, and atomic tests.

Pauly (2001) analyzes syntactic formats for safety in dynamic game logics, showing how forcing bisimulation lives in harmony with the operations of DGL.

Coda: Two levels again Bisimulation and invariance arose by thinking about internal structure of single games. But here we have seen how they also make sense for game combination. Perhaps the internal and external perspectives distinguished at the beginning of this chapter are not so different after all.

19.7 Conclusion

The main points We have explored a dynamic logic of sequential game operations merging ideas from logic games and game logics at a well-chosen abstraction level. It encodes some of the basic reasoning about strategies behind many of the systems of Part IV. The theory still looks like a propositional dynamic logic of programs, but now for complex games over neighborhood models with forcing bisimulation, and we have developed its basic model-theoretic and axiomatic properties. In addition, we found interesting new features, such as a decidable algebra of game operations.

Open problems Many questions arise now, connecting to earlier topics in this book. First, it makes sense to add imperfect information, preferences, and other features of real games, to see whether the elegant compositional design of our logic survives such an extension. Next, there is the issue of a natural operational repertoire for games, and in particular, dealing with operations for which PDL has not been so suitable historically as a process theory, including parallel game combinations. Also, much remains to be understood concerning the connection of DGL to infinite games and temporal or modal fixed point logics, a topic that will return in Chapters 20 and 25. Finally, as to game equivalences, DGL is clearly power-oriented in the sense of Chapter 11. Could there be similar logics for games at finer action levels of detail, such as the bisimulations in Chapter 1?

19.8 Literature

The presentation in this chapter is from the lecture notes of van Benthem (1999). An extension of the framework to parallel products of games is given in van Benthem et al. (2008).

Key sources on DGL are Parikh (1985) and Pauly (2001). Also relevant is the survey article of van der Hoek & Pauly (2006). Sources for general neighborhood semantics were given in Chapter 11.

19.9 Further directions

Here are two further threads following up on basic dynamic game logic. We have put them in this separate final section to avoid making the chapter top-heavy.

DGL with parallel operations Dealing with parallel composition of programs and distributed computation has not been PDL's strongest suit, and systems of process algebra took over around 1980 (cf. Milner 1999, Bergstra et al. 2001). Even so, a system of "concurrent PDL" exists using sets of local states as the output of computations (cf. Goldblatt 1992, van Benthem et al. 1994). For games, too, parallel products make sense, as we will see in Chapters 20 and 21. Indeed, standard strategic games already allowed for simultaneous moves (cf. Chapter 12).

Concurrent PDL can be merged with DGL into a natural system that performs one more set lifting (van Benthem et al. 2008). Here is a sketch. Models for concurrent PDL have accessibility relations Rs, X of type $S \times \wp(S)$ with the output read conjunctively: from state s, the program can produce all of the set X together.

DEFINITION 19.10 Game boards for concurrent *DGL*
Models for *concurrent* DGL are like models for DGL, but now with forcing relations of type $S \times \wp(\wp(S))$ running from states to families of sets of states. The interpretation of the inner set layer is as in concurrent PDL, but the outer set layer is disjunctive: the game can end in one of the sets of states collected there.[227] ■

227 This allows for two notions of monotonicity: increasing outer level sets is a form of weakening like in DGL, but increasing inner level sets means stronger output.

Our earlier forcing definitions for sequential game operations lift to this setting in a straightforward manner. But more interesting is the emergence of new operations.

DEFINITION 19.11 Forcing for parallel game product
Here is a natural *parallel product* $G \times G'$ of games, collecting outputs for subgames:

$$\rho^i_{G\times G'} sX \quad \text{iff} \quad \exists Y, Z : \rho^i_G s, Y \;\&\; \rho^i_{G'} s, Z \;\&\; X = \{y \cup z \mid y \in Y \;\&\; z \in Z\}$$

This operation fits natural games that allow for simultaneous moves, such as those of Chapters 12 and 13. The modality of the system is now interpreted as follows

$$\boldsymbol{M}, s \models \{G, i\}\varphi \quad \text{iff} \quad \exists X : \rho^{\boldsymbol{M},i}_G s, X \;\&\; \text{for all } x \in \bigcup X : M, x \models \varphi$$

Further details are as in our earlier modal logics of forcing. ■

The following theorem can be shown by a standard completeness argument.

THEOREM 19.5 The logic of the sequential game operations plus the above parallel product is axiomatizable and decidable.

The crucial DGL style decomposition axiom for the product is

$$\{G \times G', i\}\varphi \leftrightarrow \{G, i\}\varphi \wedge \{G', i\}\varphi$$

This encodes a game algebra, which has not yet been axiomatized equationally.

REMARK Collective action
The language and interpretation chosen here looks only at local properties of states. But simultaneous action is often at the same time collective action. It is also shown in van Benthem et al. (2008) how to lift the indices of evaluation to "collective set states" X instead of single points s, in a format

$$\boldsymbol{M}, X \models \{G, i\}\varphi,$$

where we can now also have a richer language with modalities referring to the natural inclusion structure of these new collective states. This modified logic of collective action for games has not yet been explored in a game setting.

Concurrent DGL seems a good fit with the IF logic of Chapter 21, although the only current study is Galliani (2012b) on connections with the dependence logics of Väänänen (2007). But quite different approaches make sense as well. Our next chapter will pursue a line on parallel games based on proof theory and linear logic.

Our second theme is logics for neighborhood models that may give the reader a better feeling for the mechanics of what has been proposed in this chapter.

PDL in neighborhood semantics The system DGL without dual is much like a propositional dynamic logic on neighborhood models with monotonic accessibility relations as in Chapter 11. It is of interest to see how its completeness works, since it ties together many notions introduced in this chapter.

THEOREM 19.6 PDL on forcing relations in neighborhood models is completely axiomatized by the modal base logic of forcing plus the standard recursive axioms of PDL for complex programs.

What follows is a compact summary of the main steps leading up to this result.

Semantics Accessibility relations for complex programs with choice and composition are defined as before, but with simplified notation, since we can now ignore player markings. The crucial further clause is for the iteration operator. It can be stated in a relational fixed point format such as that of Chapters 1, 2, and 8, now for point-to-set relations:

$$R_{\pi^*} = \mu S, x, X \bullet (x \in X \ \vee \ \exists Y(xR_\pi Y \ \wedge \ \forall y \in Y : ySX)).^{228}$$

Next, for formulas, we define the usual forcing modality:

$$\boldsymbol{M}, s \models \langle \pi \rangle \varphi \quad \text{iff} \quad \exists X : sR_\pi X \ \wedge \ \forall x \in X : \boldsymbol{M}, x \models \varphi.^{229}$$

Deduction in this system involves familiar principles from earlier in this book.

Axiomatics The logic validates all axioms of PDL minus modal distribution, now replaced by monotonicity. We still have left-distribution by the recursion axiom for program union. For iteration, we adopt the following inference principles:

$$\varphi \vee \langle \pi \rangle \langle \pi^* \rangle \varphi \to \langle \pi \rangle \varphi$$
$$\text{if } \vdash (\varphi \vee \langle \pi \rangle \alpha) \to \alpha, \text{ then } \vdash \langle \pi^* \rangle \varphi \to \alpha$$

Now we come to the proof of our main result.

---

228 While this looks like the usual notion of transitive closure, there is no guarantee that the approximation sequence stops at stage ω: finite approximations need not suffice.

229 Given the monotonicity of the relations, this is equivalent to $sR_\pi [\![\varphi]\!]^{\boldsymbol{M}}$, where the double brackets stand for the usual denotation in the model of all worlds satisfying φ.

Completeness Fix any valid formula φ, and stick to the finite sublanguage induced by it in the Fisher-Ladner closure $FL(\varphi)$ of φ (see Blackburn et al. 2001 for this standard device). Each atom s has a canonical finite description $\#s$ by enumeration, and each (finite) set of atoms X has a finite description $\#X$ by disjunction of descriptions for its members. We define the following relation for each program π:

$$sS_\pi X \text{ iff } \#s \wedge \langle\pi\rangle\#X \text{ is consistent.}$$

Taking this explanation for atomic programs only, and then proceeding inductively by the truth definition gives us the standard semantic relations R_π. The following connection between these two relations is crucial.

Inclusion Lemma $S_\pi \subseteq R_\pi$

Proof Case (a). Atomic programs a satisfy the inclusion by definition. *Case (b).* For program union, first use the PDL inference from $\langle\pi_1 \cup \pi_2\rangle\varphi$ to $\langle\pi_1\rangle\varphi \vee \langle\pi_2\rangle\varphi$ to show that, if $\#s \wedge \langle\pi_1 \cup \pi_2\rangle\#X$ is consistent, then so is $\#s \wedge \langle\pi_1\rangle\#X$ or $\#s \wedge \langle\pi_2\rangle\#X$. The rest follows by the inductive hypothesis. *Case (c).* Program composition requires a slightly more complex argument. Using one direction of the PDL composition axiom, if $\#s \wedge \langle\pi_1 ; \pi_2\rangle\#X$ is consistent, then so is $\#s \wedge \langle\pi_1\rangle\langle\pi_2\rangle\#X$. All atoms t whose $\#t$ are consistent with $\langle\pi_2\rangle\#X$ form a set Y. The following implication states how these notions behave.

Claim $\langle\pi_2\rangle\#X$ provably implies $\#Y$.[230]

The claim implies that we also have $\#s \wedge \langle\pi_1\rangle\#Y$ consistent, while, by the definition of Y, $\#t \wedge \langle\pi_2\rangle\#X$ is consistent for all $t \in Y$. But then, by the inductive hypothesis for $\pi = \pi_1 and \pi_2$, the desired inclusion $S_\pi \subseteq R_\pi$ follows. *Case (d).* The inclusion for test programs is straightforward by the inductive hypothesis. *Case (e).* Finally, dealing with program iteration crucially uses the smallest fixed point induction rule. Let $\#s \wedge \langle\pi^*\rangle\#X$ be consistent for some set of atoms X. We show that s belongs to the smallest fixed point of the following procedure. Start with X and keep applying the map

$$F(Y) = X \cup \{s \mid sS_\pi Y\}$$

230 To see this, suppose that the implication is not derivable. Then there is a maximally consistent set Σ containing $\langle\pi_2\rangle\#X$ and $\neg\#Y$. Restricting Σ to the $FL(\varphi)$ sublanguage gives an atom t in the set Y: but all of these atoms are ruled out by the presence of $\neg\#Y$. This is a contradiction, and so the claim is proved.

If we can show this fact about s, the rest of the proof follows in a standard manner, relying on the inductive hypothesis. Now, in our finite model, the given procedure stops at some finite stage in a finite set A where

$$A = X \cup \{s \mid sS_\pi A\} \qquad (\$)$$

The following claim relates this observation to a fact about provability in PDL.

CLAIM The implication $(\#X \vee \langle\pi\rangle\#A) \to \#A$ is provable.[231]

Given the implication of the claim, by the smallest fixed point rule, $\langle\pi^*\rangle\#X \to \#A$ is provable. But then, since $\#s \wedge \langle\pi^*\rangle\#X$ was consistent, $\#s \wedge \#A$ is consistent, too, and this must mean that $s \in A$. ■

The final piece of the completeness argument is the usual Truth Lemma.

TRUTH LEMMA The equivalence $\varphi \in s$ iff $s \models \varphi$ holds for all states and formulas.

Proof The proof is by induction on formulas, with a subinduction on programs for $\langle\pi\rangle\varphi$. The latter's direction from left to right uses the inclusion lemma. From right to left, the inductive steps use half of the PDL program axioms, appealing only to the monotonicity of the logic. We examine three cases. *Case (a).* Consider an atomic program a with $s \models \langle a\rangle\varphi$. By the truth definition, we have a set X with sR_aX, i.e., $\#s \wedge \langle\pi\rangle\#X$ is consistent in our logic, and for all $x \in X : x \models \varphi$. Then by the inductive hypothesis, $\varphi \in x$. Now we observe that all this implies $\vdash \#X \to \varphi$, by a standard argument about maximally consistent sets and their $FL(\varphi)$ restrictions. But then by monotonicity, $\#s \wedge \langle a\rangle\varphi$ is also consistent, and hence $\langle a\rangle\varphi \in s$. *Case (b).* Consider a program composition with $s \models \langle\pi_1 ; \pi_2\rangle\varphi$. By the truth definition, we have a set X with $sR_{\pi_1 ; \pi_2}X$ and for all $x \in X : x \models \varphi$. Thus, there is a set Y with $sR_{\pi_1}Y$ and for all $y \in Y : yR_{\pi_2}X$ while for all $x \in X : x \models \varphi$. In other words, for all $y \in Y : y \models \langle\pi_2\rangle\varphi$. Then by the inductive hypothesis, $\langle\pi_1\rangle\langle\pi_2\rangle\varphi \in s$, and using one half of the PDL axiom for composition, also $\langle\pi_1 ; \pi_2\rangle\varphi \in s$. *Case (c).* Consider a program iteration with $s \models \langle\pi^*\rangle\varphi$. Again, using the truth definition, we either φ have true at s, and hence by the inductive hypothesis in s, where it implies the presence of $\langle\pi^*\rangle\varphi$ by half of our fixed point axiom. Or, we have sets X, Y with

231 To see this, note first that $\#X \to \#A$ is provable, as $X \subseteq A$. If $\langle\pi\rangle\#A \to \#A$ is not provable, then there is a maximally consistent Σ including $\langle\pi\rangle\#A, \neg\#A$. Restricting Σ to the $FL(\varphi)$ sublanguage, we get an atom t with $tS_\pi A$ that is not in A because of the presence of $\neg\#A$ in Σ. This contradicts ($).

$sR_\pi Y \wedge \forall y \in Y : yR_{\pi^*}X$ and all $x \in X$ satisfy φ. But then all $y \in Y$ satisfy $\langle \pi^* \rangle \varphi$, and so s satisfies $\langle \pi \rangle \langle \pi^* \rangle \varphi$. Thus, by the inductive hypothesis, $\langle \pi \rangle \langle \pi^* \rangle \varphi$ is in s. By the remaining part of the fixed point axiom, we also get $\langle \pi^* \rangle \varphi$ in s. ■

The earlier open problem concerning completeness for DGL with game dual added emerges in this setting when we try to modify the above proof to deal with player-dependent accessibility relations.