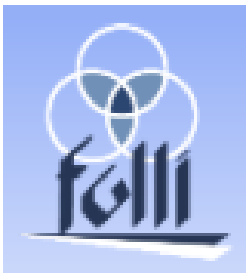
The 18th European Summer School in Logic, Language and Information

31 July - 11 August, 2006 — Málaga, Spain

The modal μ-calculus

Yde Venema

Introductory Course, Logic and Computation Section



ESSLLI is the Annual Summer School of FoLLI,
The Association for Logic, Language and Information
http://www.folli.org

Organized by

Sponsored by

Lectures on the modal μ-calculus

Yde Venema*

June 27, 2006

*Institute for Logic, Language and Computation, University of Amsterdam, Plantage Muidergracht 24, NL–1018 TV Amsterdam. E-mail: yde@science.uva.nl.

Contents

Preface

In their present incarnation, these notes serve as material for *The modal μ-calculus*, a course to be given at the ESSLLI summer school in Malaga, Spain, 2006. Earlier, they accompanied the second part of the course, *Advanced Modal Logic*, that I taught at the University of Amsterdam, 2006.

The text is written for students and researchers in logic and (theoretical) computer science who are interested in (modal) logic, and its connections with automata theory. The present version assumes some familiarity with the basic definitions of modal logic, as can be found in any recent text book.

This is very much work in progress. The final version of this text will be at least twice as long as the current one. It will contain full proofs of all results mentioned here, and additional material on for instance game theory, monadic second order logic, coalgebra (and the automata operating on them), and on other modal fixpoint logics besides the modal μ-calculus, such as fragments like CTL. Also, in the final version I hope to pay attention to more computational aspects of fixpoint logics such as model checking, and the complexity of various problems related to logic and automata theory.

The most serious shortcoming of the present text is that it contains no references, and I apologize to all the researchers that have contributed to this beautiful theory and do not find their work acknowledged here. Unfortunately, for lack of time I could not manage to provide a proper collection of notes, and hence, with hesitation, I decided better to have no references at all, than to have incomplete (and hence, unfair) ones. I hope to take care of this omission as soon as possible. Readers who want references on specific results mentioned in the text can either contact me, or consult one of the books listed in the References.

Finally, comments are appreciated very much!

Amsterdam, June 27, 2006
Yde Venema

1 Introduction

The study of the modal μ-calculus can be motivated from various (not necessarily disjoint!) directions.

Process Theory In this area of theoretical computer science, one studies formalisms for describing and reasoning about labelled transition systems — these being mathematical structures that model processes. Here the modal μ-calculus strikes a very good balance between computational efficiency and expressiveness. On the one hand, the presence of fixpoint operators make it possible to express most, if not all, of the properties that are of interest in the study of (ongoing) behavior. But on the other hand, the formalism is still simple enough to allow an (almost) polynomial model checking complexity and a exponential time satisfiability problem.

Modal Logic From the perspective of modal logic, the modal μ-calculus is a well-behaved formalism, with a great number of attractive logical properties. For instance, it is the bisimulation invariant fragment of second order logic, it enjoys uniform interpolation, and the set of its validities admits a transparent, finitary axiomatization, and has the finite model property. In short, the modal μ-calculus shares (or naturally generalizes) all the nice properties of ordinary modal logic.

Mathematics and Theoretical Computer Science More in general, the modal μ-calculus has a very interesting theory, with lots of connections with neighboring areas in mathematics and theoretical computer science. We mention automata theory (more specifically, the theory of finite automata operating on infinite objects), game theory, universal algebra and lattice theory, and the theory of universal coalgebra.

Open Problems Finally, there are still a number of interesting *open problems* concerning the modal μ-calculus. For instance, it is open whether the characterization of the modal μ-calculus as the bisimulation invariant fragment of monadic second order logic still holds if we restrict attention to finite structures, and in fact there are many open problems related to the expressiveness of the formalism. Also, the exact complexity of the model checking problem is not known. And to mention a third example: the completeness theory of modal fixpoint logics is still a largely undeveloped field.

Summarizing, the modal μ-calculus is a formalism with important applications in the field of process theory, with interesting metalogical properties, various nontrivial links with other areas in mathematics and theoretical computer science, and a number of intriguing open problems. Time to study it in more detail.

2 Modal logic

As mentioned in the preface, we assume familiarity with the basic definitions concerning the syntax and semantics of modal logic. The purpose of this first chapter is to briefly recall notation and terminology, and to provide an introduction to the coalgebraic perspective on modal logic.

2.1 Basics

Syntax

Working with fixpoint operators, we may benefit from a set-up in which the use of the negation symbol may only be applied to atomic formulas. The price that one has to pay for this is an enlarged arsenal of primitive symbols. In the context of modal logic we then arrive at the following definition.

Definition 2.1 Let P be a set of *proposition letters*, whose elements are usually denoted as $p, q, r, x, y, z, \ldots$, and let D be a set of (atomic) *actions*, whose elements are usually denoted as $d, e, c, \ldots$ The set $\mathrm{PML}_{\mathsf{D}}(\mathsf{P})$ of *Polymodal Logic* in D and P as follows:

$$\varphi \ ::= \ p \mid \neg p \mid \bot \mid \top \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \langle d \rangle \varphi \mid [d]\varphi$$

where $p \in \mathsf{P}$, and $d \in \mathsf{D}$. Elements of $\mathrm{PML}_{\mathsf{D}}(\mathsf{P})$ are called *(poly-)modal formulas*, or briefly, *formulas*. Formulas of the form p or $\neg p$ are called *literals*.

In case the set D is a singleton, we speak of the language $\mathrm{BML}(\mathsf{P})$ of *Basic Modal Logic*. $\lhd$

Usually the sets P and D are implicitly understood, and suppressed in the notation.

Remark 2.2 Generally it will suffice to treat examples, proofs, etc., from basic modal logic. $\lhd$

Remark 2.3 The *negation* $\sim\varphi$ of a formula φ can inductively be defined as follows:

$$\begin{array}{llll} \sim\bot & := \top & \sim\top & := \bot \\ \sim p & := \neg p & \sim\neg p & := p \\ \sim(\varphi \vee \psi) & := \sim\varphi \wedge \sim\psi & \sim(\varphi \wedge \psi) & := \sim\varphi \vee \sim\psi \\ \sim[d]\varphi & := \langle d \rangle \sim\varphi & \sim\langle d \rangle\varphi & := [d]\sim\varphi \end{array}$$

On the basis of this, we can also define the other standard abbreviated connectives, such as $\rightarrow$ and $\leftrightarrow$. $\lhd$

We assume that the reader is familiar with standard syntactic notions such as those of a *subformula* or the *construction tree* of a formula, and with standard syntactic operations such as *substitution*. Concerning the latter, we let $\varphi[\psi/x]$ denote the formula that we obtain by substituting all occurrences of p in φ by ψ.

Semantics

The *relational* semantics of modal logic is well known. The basic idea is that the modal operators $\langle d \rangle$ and $[d]$ are both interpreted using an *accessibility* relation R_d.

Definition 2.4 Fix a set P of proposition letters and a set D of atomic actions. A (P, D)-*transition system* or (P, D)-*Kripke model* is a triple $\mathbb{S} = \langle S, V, R \rangle$ such that S is a set of objects called *states* or *points*, $V : S \to \wp(\mathsf{P})$ is a *valuation*, and $R = \{R_d \subseteq S \times S \mid d \in \mathsf{D}\}$ is a family of binary *accessibility relations*. Elements of the set $R_d[s] := \{t \in S \mid (s,t) \in R\}$ are called *d-successors* of s. The pair (P, D) is called the *type* of the transition system.

A *pointed* transition system or Kripke model is a pair $(\mathbb{S}, s)$ consisting of a transition system $\mathbb{S}$ and a state s in $\mathbb{S}$. $\lhd$

As in the case of the syntax we will often suppress explicit reference to P and D.

The notion of truth (or satisfaction) is also defined as usual.

Definition 2.5 Let P and D be sets of proposition letters and actions, respectively, and let $\mathbb{S} = \langle S, \sigma \rangle$ be a transition system for this language. Then the *satisfaction relation* $\Vdash$ between states of $\mathbb{S}$ and formulas of PML is defined by the following formula induction.

$$\begin{array}{lll}
\mathbb{S}, s \Vdash p & \text{if} & s \in V(p), \\
\mathbb{S}, s \Vdash \neg p & \text{if} & s \notin V(p), \\
\mathbb{S}, s \Vdash \bot & & \text{never}, \\
\mathbb{S}, s \Vdash \top & & \text{always}, \\
\mathbb{S}, s \Vdash \varphi \vee \psi & \text{if} & \mathbb{S}, s \Vdash \varphi \text{ or } \mathbb{S}, s \Vdash \psi, \\
\mathbb{S}, s \Vdash \varphi \wedge \psi & \text{if} & \mathbb{S}, s \Vdash \varphi \text{ and } \mathbb{S}, s \Vdash \psi, \\
\mathbb{S}, s \Vdash \langle d \rangle \varphi & \text{if} & \mathbb{S}, t \Vdash \varphi \text{ for some } t \in R[s], \\
\mathbb{S}, s \Vdash [d] \varphi & \text{if} & \mathbb{S}, t \Vdash \varphi \text{ for all } t \in R[s].
\end{array}$$

We say that φ *is true* or *holds* at s if $\mathbb{S}, s \Vdash \varphi$, and we let the set

$$[\![\varphi]\!]^{\mathbb{S}} := \{s \in S \mid \mathbb{S}, s \Vdash \varphi\}.$$

denote the *meaning* or *extension* of φ in $\mathbb{S}$. $\lhd$

Remark 2.6 We can define semantics of modal logic in terms of $[\![\cdot]\!]$ as well. Fix an LTS $\mathbb{S}$, then define $[\![\varphi]\!]^{\mathbb{S}}$ by induction on the complexity of φ:

$$\begin{array}{lcl}
[\![p]\!]^{\mathbb{S}} & = & V(p) \\
[\![\neg p]\!]^{\mathbb{S}} & = & S \setminus V(p) \\
[\![\bot]\!]^{\mathbb{S}} & = & \varnothing \\
[\![\top]\!]^{\mathbb{S}} & = & S \\
[\![\varphi \vee \psi]\!]^{\mathbb{S}} & = & [\![\varphi]\!]^{\mathbb{S}} \cup [\![\psi]\!]^{\mathbb{S}} \\
[\![\varphi \wedge \psi]\!]^{\mathbb{S}} & = & [\![\varphi]\!]^{\mathbb{S}} \cap [\![\psi]\!]^{\mathbb{S}} \\
[\![\langle d \rangle \varphi]\!]^{\mathbb{S}} & = & \langle R_d \rangle [\![\varphi]\!]^{\mathbb{S}} \\
[\![[d] \varphi]\!]^{\mathbb{S}} & = & [R_d] [\![\varphi]\!]^{\mathbb{S}}
\end{array}$$

Here the operations $\langle R_d \rangle$ and $[R_d]$ are defined in Appendix B.

The satisfaction relation $\Vdash$ may be recovered from this by putting $\mathbb{S}, s \Vdash \varphi$ iff $s \in [\![\varphi]\!]^{\mathbb{S}}$. $\lhd$

Modal logic by now has a quite substantial model theory. One of the most fundamental notions in that theory is the following notion.

Definition 2.7 Let $\mathbb{S}$ and $\mathbb{S}'$ be two transition systems of the same type (P, D). Then a non-empty relation $Z \subseteq S \times S'$ is a *bisimulation* if the following holds, for every $(s, s') \in Z$:

(prop) $s \in V(p)$ iff $s' \in V'(p)$, for all $p \in \mathsf{P}$;

(forth) for all actions d, and for all $t \in R_d[s]$ there is a $t' \in R'_d[s']$ with $(t, t') \in Z$;

(back) for all actions d, and for all $t' \in R'_d[s']$ there is a $t \in R_d[s]$ with $(t, t') \in Z$.

Two states s and s' are called *bisimilar*, notation: $\mathbb{S}, s \leftrightarroweq \mathbb{S}', s'$ if there is some bisimulation Z with $(s, s') \in Z$.

Relations satisfying the back and forth clauses, but the (prop) clause only for a subset $\mathsf{Q} \subseteq \mathsf{P}$ are called Q-*bisimulations*, and the corresponding notion of bisimilarity is denoted by $\leftrightarroweq_{\mathsf{Q}}$. $\lhd$

- `bisimulation game`
- `bisimulation invariance of modal logic`

2.2 Game semantics

We will now describe the semantics defined above in game-theoretic terms. That is, we will define the *evaluation game* $\mathcal{E}(\xi, \mathbb{S})$ associated with a (fixed) formula ξ and a (fixed) LTS $\mathbb{S}$.

Every match of this game will be played by two *players*: Éloise ($\exists$ or 0) and Abélard ($\forall$ or 1). Given a player P, we always denote the *opponent* of P by $\bar{P}$. A *match* of the game consists of the two players moving a *token* from one position to another. That is, we are dealing with a *board game*, see Chapter 4. *Positions* are of the form (φ, s) with φ a *subformula* of ξ, and s a state of $\mathbb{S}$.

It is useful to assign *goals* to both players: in an arbitrary position (φ, s), think of $\exists$ trying to show that φ is *true* at s in $\mathbb{S}$, and of $\forall$ of trying to convince her that φ is *false* at s.

Depending on the type of the position (more precisely, on the formula part of the position), one of the two players may move the token to a next position. For instance, if the position is of the $(\langle d \rangle \varphi, s)$, it is $\exists$ who is to move, and she may choose an arbitrary

d-successor t of s where to make φ true. That is, the set of *next positions* that she may choose from is given as the set $\{(\varphi, t) \mid t \in R_d[s]\}$.

In the case there is no successor of s to choose, she immediately *loses* the game. This is a convenient way to formulate the rules for winning and losing this game: if a position (φ, s) has no admissible next positions, the player whose turn it is to play at (φ, s) immediately loses the game.

This convention gives us a nice handle on positions of the form (p, s) where p is a proposition letter: we always assign such a position an *empty* set of admissible moves, but we make $\exists$ responsible for (p, s) in case p is *false* at s, and $\forall$ in case p is *true* at s. In this way, $\exists$ immediately wins if p is true at s, and $\forall$ if it is otherwise. The rules for the negative literals $(\neg p)$ and the constants, $\bot$ and $\top$, follow a similar pattern.

The full set of rules of the game is given in Table 1. Observe that all matches of this game are finite, since at each move of the game the active formula is reduced in size. (From the general perspective of board games, this means that we need not worry about winning conditions for matches of infinite length since no such matches occur.) We may now summarize the game as follows.

Definition 2.8 Given a modal formula ξ and a transition system $\mathbb{S}$, the *evaluation game* $\mathcal{E}(\xi, \mathbb{S})$ is defined as the board game given by Table 1. The instantiation of this game with starting point (ξ, s) is denoted as $\mathcal{E}(\xi, \mathbb{S})@(\xi, s)$. $\lhd$

An *instance* of an evaluation game is a pair consisting of an evaluation game and a *starting position* of the game. Such an instance will also be called an *initialized* game, or sometimes, if no confusion is likely, simply a game.

A *strategy* for a player in an initialized game, say P, is a function telling P what to do in any match from the starting position. Such a strategy is *winning for* P if every match of the game is won by P, at least if P plays according to this strategy.

A position (φ, s) is *winning* for P if P has a winning strategy for the game initialized in that position. The set of winning positions in $\mathcal{E}(\xi, \mathbb{S})$ is denoted as $\mathrm{Win}(\mathcal{E}(\xi, \mathbb{S}))$.

The main result concerning these games is that they provide an alternative, but equivalent semantics for modal logic.

Theorem 2.9 *Let ξ be a modal formula, and let $\mathbb{S}$ be an LTS. Then for any state s in $\mathbb{S}$ it holds that*

$$(\xi, s) \in \mathrm{Win}(\mathcal{E}(\xi, \mathbb{S})) \iff \mathbb{S}, s \Vdash \xi.$$

The proof of this Theorem is straightforward and left to the reader.

2.3 Kripke coalgebras

It will be convenient to have an alternative, *coalgebraic* presentation of transition systems. We first give the formal definition, and then explain that is nothing more than an alternative presentation of the standard framework.

Position	Player	Admissible moves
$(\varphi_1 \vee \varphi_2, s)$	$\exists$	$\{(\varphi_1, s), (\varphi_2, s)\}$
$(\varphi_1 \wedge \varphi_2, s)$	$\forall$	$\{(\varphi_1, s), (\varphi_2, s)\}$
$(\langle d\rangle\varphi, s)$	$\exists$	$\{(\varphi, t) \mid t \in R_d[s]\}$
$([d]\varphi, s)$	$\forall$	$\{(\varphi, t) \mid t \in R_d[s]\}$
$(\bot, s)$	$\exists$	$\varnothing$
$(\top, s)$	$\forall$	$\varnothing$
$(p, s), s \in V(p)$	$\forall$	$\varnothing$
$(p, s), s \notin V(p)$	$\exists$	$\varnothing$
$(\neg p, s), s \notin V(p)$	$\forall$	$\varnothing$
$(\neg p, s), s \in V(p)$	$\exists$	$\varnothing$

Table 1: Evaluation game for modal logic

Definition 2.10 Fix a set P of proposition letters and a set D of atomic actions. Given a set S, let $K_{\mathsf{D},\mathsf{P}}S$ denote the set

$$K_{\mathsf{D},\mathsf{P}}S := \wp(\mathsf{P}) \times \wp(S)^{\mathsf{D}}.$$

This operation will be called the *Kripke functor* associated with D and P.

A typical element of $K_{\mathsf{D},\mathsf{P}}S$ will be denoted as (π, X), with $\pi \subseteq \mathsf{P}$ and $X = \{X_d \mid d \in \mathsf{D}\}$ with $X_d \subseteq S$ for each $d \in \mathsf{D}$. $\lhd$

We will now see that any transition systems can be presented as a pair $\mathbb{S} = \langle S, \sigma : S \to KS\rangle$ where K is the Kripke functor associated with $\mathbb{S}$. The idea is that a state s in $\mathbb{S}$ may be characterized by the pair consisting of the set of proposition letters true at s, together with, for each $d \in \mathsf{D}$, the set of R_d-successors of s.

Definition 2.11 Fix a set P of proposition letters and a set D of atomic actions. A $K_{\mathsf{D},\mathsf{P}}$-coalgebra, or *Kripke coalgebras of type* (D, P) is a pair $\mathbb{S} = \langle S, \sigma\rangle$ with $\sigma : S \to K_{\mathsf{D},\mathsf{P}}S$. $\lhd$

When we say that Kripke coalgebras are nothing more than a different *presentation* of transition systems, we feel justified by the following proposition.

Proposition 2.12 *Let* P *and* D *be sets of proposition letters and of atomic actions, respectively. Then for each set* S*, there is an isomorphism between the collection of* (P, D)*-transition systems on* S*, and the* $K_{\mathsf{D},\mathsf{P}}$*-coalgebras with carrier* S*.*

Proof. Fix a set S. To see why the proposition holds, we first make some preparations. Given that the codomain of a $K_{\mathsf{D},\mathsf{P}}$-coalgebra map is the cartesian product $\wp(\mathsf{P}) \times \wp(S)^{\mathsf{D}}$, we may clearly represent a coalgebra map $\sigma : S \to K_{\mathsf{D},\mathsf{P}}S$ as a pair of maps $\sigma_0 : S \to$

$\wp(\mathsf{P})$ and $\sigma_1 : S \to \wp(S)^{\mathsf{D}}$. On the other hand, a transition model can be seen as a structure $\langle S, V, R\rangle$ with $V : \mathsf{P} \to \wp(S)$ and $R : \mathsf{D} \to \wp(S \times S)$.

Thus it suffices to show that there is an isomorphism

$$S \to \wp(\mathsf{P}) \;\cong\; \mathsf{P} \to \wp(S) \tag{1}$$

for the 'static' part, and another one

$$S \to \wp(S)^{\mathsf{D}} \;\cong\; \mathsf{D} \to \wp(S \times S) \tag{2}$$

for the 'dynamic' side of the structures.

In both cases, the proofs are based on two key observations holding in a very general context. The first concerns the isomorphism between the sets

$$(A \times B) \to C \;\cong\; A \to (B \to C),$$

where the bijection from left to right is given by *currying* (see Appendix B). And the second observation is that

$$\wp(A) \;\cong\; A \to 2,$$

where 2 denotes the set $\{0, 1\}$, and the bijection from left to right is given by the *characteristic function*.

Now the 'static isomorphism' (1) directly follows from

$$S \to \wp(\mathsf{P}) \;\cong\; S \to (\mathsf{P} \to 2) \;\cong\; (S \times \mathsf{P}) \to 2 \;\cong\; \mathsf{P} \to (S \to 2) \;\cong\; \mathsf{P} \to \wp(S).$$

Concretely, a map $\sigma_V : S \to \wp(\mathsf{P})$ gives for each state $s \in S$ the set of proposition letters that are true at s. This provides the same information as the map V which associates with a proposition letter p the set of states in S where p holds.

Concerning the 'dynamic' part, the key observation is that any relation $R \subseteq A \times B$ can be represented as the map sending a point $a \in A$ to the collection $R[a] := \{b \in B \mid (a, b) \in R\}$ of its R-successors. In the present context of an accessibility relation R on a set S we find that

$$S \to \wp(S) \;\cong\; S \to (S \to 2) \;\cong\; (S \times S) \to 2 \;\cong\; \wp(S \times S)$$

Then, bringing the set D into the picture, we find (recall that $\wp(S)^{\mathsf{D}}$ is just another notation for $\mathsf{D} \to \wp(S)$):

$$S \to (\mathsf{D} \to \wp(S)) \;\cong\; (S \times \mathsf{D}) \to \wp(S) \;\cong\; \mathsf{D} \to (S \to \wp(S)) \;\cong\; \mathsf{D} \to \wp(S \times S).$$

This shows (2) and thus proves the proposition. QED

Convention 2.13 In the sequel, we will *identify* transition systems with their associated Kripke coalgebras. For instance, even if we represent a transition system $\mathbb{S}$ as a Kripke coalgebra $\langle S, \sigma\rangle$, we will still speak, for each action d, of the *accessibility relation* R_d of $\mathbb{S}$.

▶ `morphisms to be discussed.`

2.4 Bisimulations via relation lifting

It will be convenient to reformulate the definition of a bisimulation using the notion of *relation lifting*.

Definition 2.14 Given a relation $Z \subseteq A \times A'$, define the relation $\overline{\wp}Z \subseteq \wp A \times \wp A'$ as follows:

$$\begin{array}{rcll} \overline{\wp}Z & := & \{(X, X') \mid & \text{for all } x \in X \text{ there is an } x' \in X' \text{ with } (x, x') \in Z \\ & & \& & \text{for all } x' \in X' \text{ there is an } x \in X \text{ with } (x, x') \in Z\}. \end{array}$$

We say that $Z \subseteq A \times A'$ is *full on* A and A', notation: $Z \in A \bowtie A'$, if $(A, A') \in \overline{\wp}Z$.

Similarly, given $Z \subseteq A \times A'$, define, for a Kripke functor $K = K_{\mathsf{D},\mathsf{P}}$, the relation $\overline{K}Z \subseteq KA \times KA'$ as follows:

$$\overline{K}Z := \{((\pi, X), (\pi', X')) \mid \pi = \pi' \text{ and } (X_d, X'_d) \in \overline{\wp}Z \text{ for each } d \in \mathsf{D}\}.$$

The relations $\overline{\wp}Z$ and $\overline{K}Z$ are called the *lifting* of Z with respect to $\wp$ and K, respectively. ◁

The following characterization of bisimulations is then a straightforward consequence of the definitions.

Proposition 2.15 *Let $\mathbb{S}$ and $\mathbb{S}'$ be two K-coalgebras for some Kripke functor K, and let $Z \subseteq S \times S'$ be some relation. Then*

$$Z \text{ is a bisimulation } \textit{iff } (\sigma(s), \sigma'(s')) \in \overline{K}Z \text{ for all } (s, s') \in Z. \tag{3}$$

Using this proposition we may also give a game-theoretic characterization of the notion of bisimilarity. We first give an informal description. A match of the *bisimilarity game* between two Kripke models $\mathbb{S}$ and $\mathbb{S}'$ is played by two players, $\exists$ and $\forall$. As in the evaluation game, these players move a token around from one *position* of the game to another.

In the game there are two kinds of positions: pairs of the form $(s, s') \in S \times S'$ are called *basic positions* and belong to $\exists$. The other positions are of the form $Z \subseteq S \times S'$ and belong to $\forall$.

The idea of the game is that at a position (s, s'), $\exists$ claims that s and s' are bisimilar, and to substantiate this claim she proposes a *local bisimulation*, that is, a relation $Z \subseteq S \times S'$ with $(\sigma(s), \sigma'(s')) \in \overline{K}Z$. Implicitly, her claim at a position $Z \subseteq S \times S'$ is that *all* pairs in Z are bisimilar, so $\forall$ can pick an arbitrary pair $(t, t') \in Z$ and challenge $\exists$ to show that this t and t' are bisimilar.

If a player gets stuck in a match of this game, then their opponent wins the match. For instance, if s and s' disagree about the truth of some proposition letter, then *no* relation $Z \subseteq S \times S'$ will satisfy that $(\sigma(s), \sigma'(s')) \in \overline{K}Z$. Hence such positions are an

immediate loss for $\exists$. But on the other hand, if neither s nor s' has successors, and agree on the truth of all proposition letters, then $\exists$ could choose the *empty* relation as a local bisimulation, and $\forall$ would lose the match at his next move.

A new option arises if neither player gets stuck: this game may also have matches that last *forever*. Nevertheless, we can still declare a winner for such matches, and the agreement is that $\exists$ is the winner of any infinite match. Formally, we put the following.

Definition 2.16 The *bisimilarity game* $\mathcal{B}(\mathbb{S}, \mathbb{S}')$ between two Kripke models $\mathbb{S}$ and $\mathbb{S}'$ is given as the board game defined by Table 2, where the winning condition for infinite matches is simply that $\exists$ wins all infinite matches. $\lhd$

Position	Player	Admissible moves
$(s, s') \in S \times S'$	$\exists$	$\{Z \in \wp(S \times S') \mid (\sigma(s), \sigma'(s')) \in \overline{K}Z\}$
$Z \in \wp(S \times S')$	$\forall$	Z

Table 2: Bisimilarity game for Kripke models

The following theorem states that the collection of basic winning positions for $\exists$ forms the *largest bisimulation* between $\mathbb{S}$ and $\mathbb{S}'$.

Theorem 2.17 *Let* $(\mathbb{S}, s)$ *and* $(\mathbb{S}', s')$ *be two pointed Kripke models. Then* $\mathbb{S}, s \leftrightarrows \mathbb{S}, s'$ *iff* $(s, s') \in \mathrm{Win}_{\exists}(\mathcal{B}(\mathbb{S}, \mathbb{S}'))$.

Proof. For the direction from left to right: suppose that Z is a bisimulation between $\mathbb{S}$ and $\mathbb{S}'$ linking s and s'. Suppose that $\exists$, starting from position (s, s'), always chooses the relation Z itself as the local bisimulation. It is then easy to verify, by induction on the length of the match, that this strategy always provides her with a legitimate move, and that it keeps her alive forever. This shows that it is a winning strategy.

For the converse direction, it suffices to show that the relation $\mathrm{Win}_{\exists}(\mathcal{B}(\mathbb{S}, \mathbb{S}'))$ itself is in fact a bisimulation. The proof of this is straightforward, and we leave the details for the reader. QED

2.5 The cover modality

As we will see now, there is an interesting alternative for the standard formulation of basic modal logic in terms of boxes and diamonds. This alternative set-up is based on a connective which turns *sets* of formulas into formulas.

Definition 2.18 Let Φ be a finite set of formulas. Then $\nabla\Phi$ is a formula, which holds at a state s in a Kripke model if *every* formula in Φ holds at *some* successor of s, while at the same time, *every* successor of s makes *some* formula in Φ true. The operator ∇ is called the *cover modality*. $\lhd$

Observe that this definition involves the ∀∃&∀∃ pattern that we know from the notion of *relation lifting* $\overline{\wp}$ defined in the previous section. In other words, the semantics of the cover modality can be expressed in terms of relation lifting. For that purpose, observe that we may think of the forcing or satisfaction relation $\Vdash$ simply as a binary relation between states and formulas.

Proposition 2.19 *Let s be some state in a Kripke model $\mathbb{S}$, and let Φ be a set of formulas. Then*

$$\mathbb{S}, s \Vdash \nabla\Phi \text{ iff } (\sigma(s), \Phi) \in \overline{\wp}(\Vdash).$$

Proof. Immediate by unravelling the definitions. QED

It is not so hard to see that the cover modality can be defined in the standard modal language:

$$\nabla\Phi \equiv \Box \bigvee \Phi \wedge \bigwedge \Diamond\Phi, \tag{4}$$

where $\Diamond\Phi$ denotes the set $\{\Diamond\varphi \mid \varphi \in \Phi\}$.

Things start to get interesting once we realize that *both* the ordinary diamond $\Diamond$ *and* the ordinary box $\Box$ can be expressed in terms of the cover modality (and the disjunction):

$$\begin{array}{lcl} \Diamond\varphi & \equiv & \nabla\{\varphi, \top\}, \\ \Box\varphi & \equiv & \nabla\varnothing \vee \nabla\{\varphi\}. \end{array} \tag{5}$$

Here, as always, we use the convention that $\bigvee \varnothing = \bot$ and $\bigwedge \varnothing = \top$.

Making the above observations more precise, we arrive at the following definition and proposition.

Definition 2.20 Formulas of the language BML_∇ are given by the following recursive definition:

$$\varphi \ ::= \ p \mid \neg p \mid \bot \mid \top \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \nabla\Phi$$

where Φ denotes a set of formulas. ◁

Proposition 2.21 *The languages* BML *and* BML_∇ *are equally expressive.*

Proof. Immediate by (4) and (5). QED

The *real* importance of the cover modality is that it allows us to almost completely eliminate the Boolean *conjunction*. This remarkable fact is based on the following distributive law. Recall from Definition 2.14 that we write $Z \in A \bowtie A'$ if a relation $Z \subseteq A \times A'$ is *full on* A and A', that is, if $(A, A') \in \overline{\wp}Z$.

Proposition 2.22 *For all pairs Φ, Φ' of sets of formulas, the following two formulas are equivalent:*

$$\nabla\Phi \wedge \nabla\Phi' \;\equiv \bigvee_{Z \in \Phi \bowtie \Phi'} \nabla\{\varphi \wedge \varphi' \mid (\varphi, \varphi') \in Z\}. \tag{6}$$

Proof. For the direction from left to right, suppose that $\mathbb{S}, s \Vdash \nabla\Phi \wedge \nabla\Phi'$. Let $Z \subseteq \Phi \times \Phi'$ consist of those pairs (φ, φ') such that the conjunction $\varphi \wedge \varphi'$ is true at some successor t of s. It is then straightforward to verify that Z is full on Φ and Φ', and that $\mathbb{S}, s \Vdash \nabla\{\varphi \wedge \varphi' \mid (\varphi, \varphi') \in Z\}$.

The converse direction follows fairly directly from the definitions. QED

2.6 Coalgebraic modal logic

Using the cover modality introduced in the previous section, we can show that we can restrict the use of conjunction in modal logic to that of the *special conjunction* connective $\bullet$. First however, we take care of the proposition letters.

Definition 2.23 Fix a set P of proposition letters. Given a subset $\pi \subseteq \mathsf{P}$, we let $\odot\pi$ denote the formula with semantics given by

$$\mathbb{S}, s \Vdash \odot\pi \text{ iff } \sigma_V(s) = \pi$$

for any $K_{\mathsf{D},\mathsf{P}}$-coalgebra $\mathbb{S} = \langle S, \sigma \rangle$. $\lhd$

In words, the formula $\odot\pi$ holds at a state s iff π consists precisely of those proposition letters in P that are true at s, or equivalently,

$$\odot\pi := \bigwedge_{p \in \pi} p \wedge \bigwedge_{p \notin \pi} \neg p.$$

It is not difficult to see that every propositional formula with proposition letters from P can be expressed as disjunctions of formulas of the form $\odot\pi$. In particular, it is straightforward to verify that

$$q \;\equiv \bigvee_{q \in \pi} \odot\pi$$

for every $q \in \mathsf{P}$.

We are now ready for the introduction of the coalgebraic modal connective $\bullet$.

Definition 2.24 Fix sets P of proposition letters and D of atomic actions, respectively. Given a subset $\pi \subseteq \mathsf{P}$, and a D-indexed family $\Phi = \{\Phi_d \mid d \in \mathsf{D}\}$ of formulas, then $\pi \bullet \Phi$ is a formula, of which the semantics is defined by the following equivalence:

$$\pi \bullet \Phi \;\equiv\; \odot\pi \wedge \bigwedge_{d \in \mathsf{D}} \nabla_d \Phi_d.$$

Here ∇_d is the cover modality associated with the accessibility relation R_d of d.

The set $\mathrm{CML}_{\mathsf{D}}(\mathsf{P})$ of *coalgebraic modal formulas* is given as follows:

$$\varphi \ ::= \ \bot \ \mid \ \top \ \mid \ \varphi \vee \varphi \ \mid \ \varphi \wedge \varphi \ \mid \ \pi \bullet \Phi.$$

$\lhd$

In words, $\pi \bullet \Phi$ is the conjunction of (i) a complete description of the *local* situation in terms of the proposition letters being true or false, and (ii) for each action d, a description of the d-successor set of the current state, using the cover modality for R_d.

The bullet modality and the language CML built on it, are called 'coalgebraic' for two reasons. First, we may identify the formula $\pi \bullet \Phi$ with the pair (π, Φ) belonging to the set $K_{\mathsf{P},\mathsf{D}}(\mathrm{CML}_{\mathsf{D}}(\mathsf{P}))$. But more importantly, the point is that the *semantics* of the bullet modality can be expressed in coalgebraic terms of relation lifting with respect to the associated Kripke functor, witness the following proposition. Note that for any transition system $\mathbb{S}$ of type K, the satisfaction relation $\Vdash \subseteq S \times \mathrm{CML}$ can be lifted to a relation $\overline{K}(\Vdash) \subseteq K(S) \times K(\mathrm{CML})$.

Proposition 2.25 *Fix sets* P *of proposition letters and* D *of atomic actions, respectively. Then we have*

$$\mathbb{S}, s \Vdash \pi \bullet \Phi \ \textit{iff} \ (\sigma(s), (\pi, \Phi)) \in \overline{K}(\Vdash) \tag{7}$$

for any $\pi \in \wp(\mathsf{P})$ *any* D*-indexed family* $\Phi = \{\Phi_d \mid d \in \mathsf{D}\}$*, and for any pointed transition system* $(\mathbb{S}, s)$ *of type* (P, D)*, the following equivalence holds.*

Proof. Simply spell out the definitions. QED

In fact, we could have taken (7) below as the *definition* of the semantics of the bullet modality.

The following result is not very hard to prove.

Theorem 2.26 *For any* P *and* D*, the languages* $\mathrm{PML}_{\mathsf{D}}(\mathsf{P})$ *and* $\mathrm{CML}_{\mathsf{D}}(\mathsf{P})$ *are expressively equivalent.*

Proof. There is a straightforward translation from CML-formulas to ordinary modal formulas, so we focus on the other direction.

It is not hard to verify that every polymodal formula can be rewritten to an equivalent formula using the connectives $\top, \bot, \wedge, \vee, \odot$ and ∇_d. But then the proof of the theorem is straightforward by the observation that both $\odot\pi$ and $\nabla_d\varphi$ can be rewritten to formulas using the bullet connective, using the equivalences below:

$$\begin{array}{rcl} \top & \equiv & \nabla_d\varnothing \vee \nabla_d\{\top\} \\ \top & \equiv & \bigvee_{\pi \subseteq \mathsf{P}} \odot\pi. \end{array}$$

For instance, this allows us to write

$$\begin{array}{rcl} \odot\pi & \equiv & \odot\pi \wedge \bigwedge_{d\in\mathsf{D}} \top \\ & \equiv & \odot\pi \wedge \bigwedge_{d}(\nabla_d\varnothing \vee \nabla_d\{\top\}) \\ & \equiv & \bigvee_{\Phi:\mathsf{D}\to\{\varnothing,\{\top\}\}} \pi \bullet \{\Phi(d) \mid d \in \mathsf{D}\}. \end{array}$$

QED

It may come as a surprise to the reader that the bullet operator is in fact the *only* form of conjunction that we need! More precisely, Theorem 2.28 below states that every formula of CML can be rewritten into an equivalent version that does not use the ordinary Boolean conjunction, but only the special 'bullet conjunction'.

Definition 2.27 Formulas of the language $\mathrm{CML}^-_{\mathsf{D}}(\mathsf{P})$ are given by the following recursive definition:

$$\varphi \;::=\; \top \mid \bot \mid \varphi \vee \varphi \mid \pi \bullet \Phi$$

where π denotes a subset of P, and Φ a D-indexed set of $\mathrm{CML}^-_{\mathsf{D}}(\mathsf{P})$-formulas. $\lhd$

Theorem 2.28 *For any* P *and* D*, the languages* $\mathrm{PML}_{\mathsf{D}}(\mathsf{P})$ *and* $\mathrm{CML}^-_{\mathsf{D}}(\mathsf{P})$ *are expressively equivalent.*

Proof. Obviously it suffices to prove that every CML-formula φ has an equivalent formula φ' that does not use the conjunction symbol. We will prove this result by formula induction, confining ourselves to the case of basic modal logic (with one action).

In the base step of this induction there is nothing to prove. In the inductive step, the clauses for the disjunction and the cover modality speak for themselves:

$$\begin{array}{rcl} \overline{\varphi \vee \psi} & := & \overline{\varphi} \vee \overline{\psi}, \\ \overline{\pi \bullet \Phi} & := & \bigvee_{\pi\subseteq\mathsf{P}} \pi \bullet \Phi. \end{array}$$

This leaves the case of a conjunction $\varphi\wedge\varphi'$, where we make a further case distinction. If either of the formulas is of the form $\top$ or $\bot$ it is obvious how to proceed: $\overline{\bot \wedge \varphi} := \bot$, $\overline{\top \wedge \varphi} := \overline{\varphi}$, etc. Also, in case either of the two conjuncts is a disjunction, say $\varphi = \varphi_0 \vee \varphi_1$, using induction loading we may correctly define $\overline{\varphi \wedge \varphi'} := \overline{\varphi_0 \wedge \varphi'} \vee \overline{\varphi_0 \wedge \varphi'}$.

The heart of the proof lies in the one remaining inductive case, namely, where $\varphi = \pi \bullet \Phi$ and $\varphi' = \pi' \bullet \Phi'$. Here we put

$$\overline{\varphi \wedge \varphi'} := \begin{cases} \bot & \text{if } \pi \neq \pi', \\ \bigvee_{Z\in\Phi\bowtie\Phi'}(\pi \bullet \{\overline{\varphi \wedge \varphi'} \mid (\varphi,\varphi') \in Z\}) & \text{if } \pi \neq \pi'. \end{cases}$$

It then follows immediately from the inductive assumptions that $\overline{\varphi \wedge \varphi'}$ is a BML^-_{∇}-formula, and from Proposition 2.22 that $\overline{\varphi \wedge \varphi'}$ is equivalent to $\varphi \wedge \varphi'$. QED

3 The modal μ-calculus

This chapter is a first introduction to the modal μ-calculus. We first introduce the language and some syntactic issues, and then proceed to the semantics. In fact we will provide *two* kinds of semantics: first the fixpoint semantics, and then the game-theoretic approach. We then show that these two approaches are equivalent. First however, we give an example of a fixpoint formula.

Example 3.1 Consider the formula $\langle a^* \rangle p$ from propositional dynamic logic. By definition, this formula holds at those points in an LTS $\mathbb{S}$ from which there is a finite path, of unspecified length, leading to a state where p is true — in our notation (see Appendix B) this set is denoted as $\langle R_a^* \rangle V(p)$.

It is then easy to see, and left to the reader as an exercise, that

$$\mathbb{S} \Vdash \langle a^* \rangle p \leftrightarrow (p \vee \langle d \rangle \langle a^* \rangle p)$$

Informally, one might say that $\langle a^* \rangle p$ is a *fixed point* or solution of the 'equation' $x \leftrightarrow p \vee \langle d \rangle x$.

Making this intuition more precise, we have to look at the formula $\delta = p \vee \langle d \rangle x$ as an operation. The idea is that the value (that is, the extension) of this formula is a function of the value of x, provided that we keep the value of p constant. Varying the value of x boils down to considering 'x-variants' of the valuation V of $\mathbb{S} = \langle S, R, V \rangle$. Let, for $X \subseteq S$, $V[x \mapsto X]$ denote the valuation that is exactly like V apart from mapping x to X, and let $\mathbb{S}[x \mapsto X]$ denote the x-variant $\langle S, R, V[x \mapsto X] \rangle$ of $\mathbb{S}$. Then $[\![\delta]\!]^{\mathbb{S}[x \mapsto X]}$ denotes the extension of δ in this x-variant. It follows from this that the formula δ *induces* the following function $\delta_x^{\mathbb{S}}$ on the power set of S:

$$\delta_x^{\mathbb{S}}(X) := [\![\delta]\!]^{\mathbb{S}[x \mapsto X]}.$$

Now we can make precise in which sense $\langle a^* \rangle p$ is a fixpoint formula: it is simply because its extension, that is, the set $\langle R_a^* \rangle V(p)$, is a fixpoint of this map $\delta_x^{\mathbb{S}}$:

$$\delta_x^{\mathbb{S}}([\![\langle a^* \rangle p]\!]^{\mathbb{S}}) = [\![\langle a^* \rangle p]\!]^{\mathbb{S}}.$$

One may show that $\langle R_a^* \rangle V(p)$ is not the *only* fixpoint of the map $\delta_x^{\mathbb{S}}$. Let $dom(R_a^\infty)$ denote the set of points from which some infinite path emanates. Then we leave it as an exercise to the reader that the set

$$\langle R_a^* \rangle V(p) \cup dom(R_a^\infty)$$

is another fixpoint of δ.

In fact, one may prove that the two mentioned fixpoints are the smallest and largest possible solutions of the equation $\delta_x^{\mathbb{S}}(X) = X$, respectively.

▶ `evaluation game for this formula: unfolding of` x ◁

As we will see in this section, the modal μ-calculus allows one to explicitly refer to such smallest and largest solutions. For instance, the formula $\langle a^* \rangle p$ of propositional dynamic logic can be written as $\mu x.p \vee \langle d \rangle x$.

3.1 Syntax

In the case of fixpoint formulas we will usually work with formulas in *positive normal form* in which the only admissible occurrences of the negation symbol is in front of atomic formulas. As always, the price that we have to pay for this is an enlarged repertoire of primitive symbols.

Definition 3.2 Given sets P and D of proposition letters and atomic actions, respectively, define the collection $\mu\mathrm{PML}(\mathsf{D}, \mathsf{P})$ of *(poly-)modal fixpoint formulas* as follows:

$$\varphi \ ::= \ \top \mid \bot \mid p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \langle d \rangle \varphi \mid [d]\varphi \mid \mu x.\varphi \mid \nu x.\varphi$$

where $p, x \in \mathsf{P}$, $a \in \mathsf{D}$, and in $\mu x.\varphi$, x may not occur in φ in a subformula $\neg x$.

As before, we will usually write $\mu\mathrm{PML}$ rather than $\mu\mathrm{PML}(\mathsf{D}, \mathsf{P})$ in order not to clutter up notation. In case the set D of atomic actions is a singleton, we will simply speak of the *modal μ-calculus*, notation: $\mu\mathrm{ML}(\mathsf{P})$, or $\mu\mathrm{ML}$ if P is understood.

The syntactic combinations μx and νx are called the *least* and *greatest fixpoint operators*, respectively. We use the symbol η to denote either μ or ν. A fixpoint formula of the form $\mu x.\varphi$ is called a *μ-formula*, while *ν-formulas* are the ones of the form $\nu x.\varphi$. $\lhd$

Definition 3.3 The concepts of *subformula* and *proper subformula* are defined as usual. We write $\varphi \trianglelefteq \psi$ if φ is a subformula of ψ. The set of subformulas of ψ is denoted as $\mathit{Sfor}(\psi)$. $\lhd$

Syntactically, the fixpoint operators are very similar to quantifiers in the way they *bind* variables.

Definition 3.4 Fix a formula φ. The sets $FV(\varphi)$ and $BV(\varphi)$ of *free* and *bound variables* of φ are defined by the following induction on φ:

$FV(\bot)$	$:=$	$\varnothing$	$BV(\bot)$	$:=$	$\varnothing$
$FV(\top)$	$:=$	$\varnothing$	$BV(\top)$	$:=$	$\varnothing$
$FV(p)$	$:=$	$\{p\}$	$BV(p)$	$:=$	$\varnothing$
$FV(\neg p)$	$:=$	$\{p\}$	$BV(\neg p)$	$:=$	$\varnothing$
$FV(\varphi \vee \psi)$	$:=$	$FV(\varphi) \cup FV(\psi)$	$BV(\varphi \vee \psi)$	$:=$	$BV(\varphi) \cup BV(\psi)$
$FV(\varphi \wedge \psi)$	$:=$	$FV(\varphi) \cup FV(\psi)$	$BV(\varphi \wedge \psi)$	$:=$	$BV(\varphi) \cup BV(\psi)$
$FV(\langle d \rangle \varphi)$	$:=$	$FV(\varphi)$	$BV(\langle d \rangle \varphi)$	$:=$	$BV(\varphi)$
$FV([d]\varphi)$	$:=$	$FV(\varphi)$	$BV([d]\varphi)$	$:=$	$BV(\varphi)$
$FV(\mu x.\varphi)$	$:=$	$FV(\varphi) \setminus \{x\}$	$BV(\mu x.\varphi)$	$:=$	$BV(\varphi) \cup \{x\}$

$\lhd$

Formulas like $x \vee \mu x.((p \vee x) \wedge \Box \nu x.\Diamond x)$ may be well formed, in practice they are unreadable. In the sequel we will almost exclusively work with formulas in which every bound variable uniquely determines a fixpoint operator binding it, and in which there is no overlap between free and bound variables.

Definition 3.5 A formula $\varphi \in \mu\mathrm{PML}$ is *clean* if no two distinct (occurrences of) fixed point operators in φ bind the same variable, and no variable has both free and bound occurrences in φ. If x is a bound variable of the clean formula φ, we let $\varphi_x = \eta_x x.\delta_x$ denote the unique subformula of φ where x is bound by the fixpoint operator η_x.

Given a clean formula φ, we define a *dependency order* on the set $BV(\varphi)$, saying that y *ranks higher* than x, notation: $x \leq_\varphi y$ iff $\varphi_x \trianglelefteq \varphi_y$. $\lhd$

The idea behind the dependency order is that if $x \leq y$, the meaning of φ_x is (in principle) dependent on the meaning of y, because y may occur freely in φ_x.

Definition 3.6 A variable x is *guarded* in a μPML-formula φ if every occurrence of x in φ is in the scope of a modal operator. A formula $\xi \in \mu\mathrm{PML}$ is *guarded* if for every subformula of ξ of the form $\eta x.\delta$, x is guarded in δ. $\lhd$

3.2 Semantics

The intended semantics of $\mu x.\varphi$ in an LTS $\mathbb{S}$ is as a *least fixpoint* (see Appendix A for the basics of fixpoint theory). For this purpose, we will consider the formula φ as an *operation* on the power set of (the state space of) $\mathbb{S}$, and we have to prove that this operation indeed has a least fixpoint. Basically, as in Example 3.1, the idea is that the meaning of the formula φ in $\mathbb{S}$ is a function of the meaning of x. In order to make this precise, we need some preliminary definitions.

Definition 3.7 Given an LTS $\mathbb{S} = \langle S, V, R\rangle$ and subset $X \subseteq S$, define the valuation $V[x \mapsto X]$ by putting

$$V[x \mapsto X](y) \;:=\; \begin{cases} V(y) & \text{if } y \neq x, \\ X & \text{if } y = x. \end{cases}$$

Then, the LTS $\mathbb{S}[x \mapsto X]$ is given as the structure $\langle S, V[x \mapsto X], R\rangle$. $\lhd$

Now inductively assume that $[\![\varphi]\!]^{\mathbb{S}}$ has been defined for all LTSs. Given a labelled transition system $\mathbb{S}$ and a propositional variable $x \in \mathsf{P}$, each formula φ induces a map $\varphi_x^{\mathbb{S}} : \wp(S) \to \wp(S)$ defined by

$$\varphi_x^{\mathbb{S}}(X) := [\![\varphi]\!]^{\mathbb{S}[x \mapsto X]}$$

Example 3.8 a) Where $\varphi_a = p \vee x$ we have $(\varphi_a)^{\mathbb{S}}_x(X) = [\![p \vee x]\!]^{\mathbb{S}[x \mapsto X]} = V(p) \cup X$.
b) Where $\varphi_b = \neg x$ we have $(\varphi_b)^{\mathbb{S}}_x(X) = [\![\neg x]\!]^{\mathbb{S}[x \mapsto X]} = S \setminus X$.
c) Where $\varphi_c = p \vee \langle d \rangle x$ we find $(\varphi_c)^{\mathbb{S}}_x(X) = [\![p \vee \langle d \rangle x]\!]^{\mathbb{S}[x \mapsto X]} = V(p) \cup \langle R_a \rangle X$.
d) Where $\varphi_d = \langle d \rangle \neg x$ we find $(\varphi_d)^{\mathbb{S}}_x(X) = [\![\langle d \rangle \neg x]\!]^{\mathbb{S}[x \mapsto X]} = \langle R_a \rangle (S \setminus X)$. ◁

Recall from the basic fixpoint theory sketched in Appendix A that a *fixpoint* of the map $\varphi^{\mathbb{S}}_x$ is a subset $X \subseteq S$ such that $\varphi^{\mathbb{S}}_x(X) = X$. Alternatively but equivalently, X is a fixpoint of $\varphi^{\mathbb{S}}_x$ iff $\mathbb{S}[x \mapsto X] \Vdash x \leftrightarrow \varphi$. Likewise, X is a *prefixpoint* of $\varphi^{\mathbb{S}}_x$ iff $\mathbb{S}[x \mapsto X] \Vdash \varphi \to x$, and a *postfixpoint* of $\varphi^{\mathbb{S}}_x$ iff $\mathbb{S}[x \mapsto X] \Vdash x \to \varphi$.

Example 3.9 Consider the formulas of Example 3.8.
a) The sets $V(p)$ and S are fixpoints of φ_a, as is in fact any X with $V(p) \subseteq X \subseteq S$.
b) Since we do not consider structures with empty domain, the formula $\neg x$ has no fixpoints at all.
c) Two fixpoints of φ_c were already given in Example 3.1.
d) Consider any model $\mathbb{Z} = \langle Z, S, V \rangle$ based on the set Z of integers, where $S = \{(z, z+1) \mid z \in Z\}$ is the successor relation. Then the only two fixpoints of φ_d are the sets of even and odd numbers, respectively. ◁

In particular, it is not the case that every formula has a least fixpoint. However, if we can guarantee that the induced function $\varphi^{\mathbb{S}}_x$ of φ is monotone, then the Knaster-Tarski theorem (Theorem A.5) provides both least and greatest fixpoints of $\varphi^{\mathbb{S}}_x$. But this is the reason why in the definition of fixpoint formulas, there is a condition in the clauses for $\eta x.\varphi$, namely that x may not occur in a subformula of the form $\neg x$. This condition on x guarantees monotonicity of the function $\varphi^{\mathbb{S}}_x$.

Definition 3.10 Given a μPML-formula φ and a labelled transition system $\mathbb{S} = \langle S, V, R \rangle$, we define the *meaning* $[\![\varphi]\!]^{\mathbb{S}}$ of φ in $\mathbb{S}$, by the following formula induction:

$$\begin{array}{lcl}
[\![\bot]\!]^{\mathbb{S}} & = & \varnothing \\
[\![\top]\!]^{\mathbb{S}} & = & S \\
[\![p]\!]^{\mathbb{S}} & = & V(p) \\
[\![\neg p]\!]^{\mathbb{S}} & = & S \setminus V(p) \\
[\![\varphi \vee \psi]\!]^{\mathbb{S}} & = & [\![\varphi]\!]^{\mathbb{S}} \cup [\![\psi]\!]^{\mathbb{S}} \\
[\![\varphi \wedge \psi]\!]^{\mathbb{S}} & = & [\![\varphi]\!]^{\mathbb{S}} \cap [\![\psi]\!]^{\mathbb{S}} \\
[\![\langle d \rangle \varphi]\!]^{\mathbb{S}} & = & \langle R_a \rangle [\![\varphi]\!]^{\mathbb{S}} \\
[\![[d]\varphi]\!]^{\mathbb{S}} & = & [R_a] [\![\varphi]\!]^{\mathbb{S}} \\
[\![\mu x.\varphi]\!]^{\mathbb{S}} & = & \bigcap \mathrm{PRE}(\varphi^{\mathbb{S}}_x) \\
[\![\nu x.\varphi]\!]^{\mathbb{S}} & = & \bigcup \mathrm{POS}(\varphi^{\mathbb{S}}_x)
\end{array}$$

Here the map $\varphi^{\mathbb{S}}_x$, for $x \in \mathsf{P}$, is (inductively) given as $\varphi^{\mathbb{S}}_x(X) = [\![\varphi]\!]^{\mathbb{S}[x \mapsto X]}$.
We write $\mathbb{S}, s \Vdash \varphi$ in case $s \in [\![\varphi]\!]^{\mathbb{S}}$. ◁

Theorem 3.11 *Let φ be an μPML-formula, in which x occurs only positively, and let $\mathbb{S}$ be a labelled transition system. Then $[\![\mu x.\varphi]\!]^{\mathbb{S}} = \mathrm{LFP}.\varphi_x^{\mathbb{S}}$, and $[\![\nu x.\varphi]\!]^{\mathbb{S}} = \mathrm{GFP}.\varphi_x^{\mathbb{S}}$.*

Proof. This is an immediate consequence of the Knaster-Tarski theorem, provided we can prove that $\varphi_x^{\mathbb{S}}$ is monotone in x if all occurrences of x in φ are positive.

▶ `Details to be supplied`

QED

It follows from the definitions that the set μPML is closed under taking *negations*.

Definition 3.12 Given a modal fixpoint formula φ, define $\sim\varphi$ inductively as follows:

$$\begin{array}{lcllcl}
\sim\bot & := & \top & \sim\top & := & \bot \\
\sim\neg p & := & p & \sim p & := & \neg p \\
\sim\varphi \vee \psi & := & \sim\varphi \wedge \sim\psi & \sim\varphi \wedge \psi & := & \sim\varphi \vee \sim\psi \\
\sim[d]\varphi & := & \langle d\rangle\sim\varphi & \sim\langle d\rangle\varphi & := & [d]\sim\varphi \\
\sim\mu x.\varphi & := & \nu x.\sim\varphi[x/\neg x] & \sim\nu x.\varphi & := & \mu x.\sim\varphi[x/\neg x]
\end{array}$$

Here $\varphi[x/\neg x]$ denotes the formula φ with (note!) all occurrences of $\neg x$ replaced with x. ◁

As an example, the reader is invited to check that $\sim(\mu x.p \vee \Diamond x) = \nu x.\neg p \wedge \Box x$.

Perhaps the clause for the fixpoint operators requires some explanation. Consider for instance the case of $\sim\mu x.\varphi$. In the proof of Proposition 3.13 we explain why our definition is the right one, here we only argue that at least it produces a well-formed modal fixpoint formula. First observe that since in φ no x occurs in a subformula $\neg x$, in $\sim\varphi$ *all* occurrences of x are negated. Hence, if we replace every occurrence of $\neg x$ with x, we again obtain a formula in which no occurrence of x is below a negation sign, and hence, we may legitimately put a fixpoint operator in front of $\sim\varphi[\neg x/x]$. Note that the net effect of these syntactic transformations is that the *bound* variables of a fixpoint formula remain unchanged.

Informally then, $\sim\varphi$ is the result of simultaneously replacing all occurrences of $\top$ with $\bot$, of p with $\neg p$ (for *free* variables p), of $\wedge$ with $\vee$, of $[d]$ with $\langle d\rangle$, of μx with νx, and vice versa, while leaving occurrences of bound variables unchanged.

Proposition 3.13 *Let φ be a modal fixpoint formula. Then $\sim\varphi$ corresponds to the negation of φ, that is,*

$$[\![\sim\varphi]\!]^{\mathbb{S}} = S \setminus [\![\varphi]\!]^{\mathbb{S}}$$

for every labelled transition system $\mathbb{S}$.

Proof. We prove this proposition by induction on the complexity of φ. We only consider the inductive case of the fixpoint operators. Leaving all other cases as exercises for the reader, we concentrate on the inductive case where φ is of the form $\mu x.\psi$.

The point is the following. Given a monotone map $f : \wp(S) \to \wp(S)$, the complement of the least fixpoint of f is equal to the greatest fixpoint of the *dual* map $\tilde{f}$ of f, given by $\tilde{f}(X) = {\sim_S} f({\sim_S} X)$, cf. Proposition A.13. (Here ${\sim_S} X = S \setminus X$ denotes the complement of X in S). Thus we are done if we can show that

$$\widetilde{\psi^{\mathbb{S}}_x} \;=\; (\sim\psi[x/\neg x])^{\mathbb{S}}_x. \tag{8}$$

But in the case that $f = \psi^{\mathbb{S}}_x$, inductively we have that $\tilde{f}(X) = {\sim_S}\psi^{\mathbb{S}}_x({\sim_S} X) = (\sim\psi)^{\mathbb{S}}_x({\sim_S} X)$. Since all occurrences of x in $\sim\psi$ are inside a subformula $\neg x$, it follows that $(\sim\psi)^{\mathbb{S}}_x({\sim_S} X) = (\sim\psi[x/\neg x])^{\mathbb{S}}_x(X)$. This proves (8). QED

Remark 3.14 It follows from the Proposition above that we could indeed have defined the language of the modal μ-calculus with a far more parsimonious supply of primitives. Given sets P and D of proposition letters and atomic actions, respectively, we could have defined the set of modal fixpoint formulas using the following induction:

$$\varphi \;::=\; \top \;\mid\; p \;\mid\; \neg\varphi \;\mid\; \varphi \vee \varphi \;\mid\; \langle d\rangle\varphi \;\mid\; \mu x.\varphi$$

where $p, x \in \mathsf{P}$, $a \in \mathsf{D}$, and in $\mu x.\varphi$, all free occurrences of x must be positive (that is, under an even number of negation symbols). Here we define $FV(\neg\varphi) = FV(\varphi)$ and $BV(\neg\varphi) = BV(\varphi)$.

In this set-up, the connectives $\wedge$ and $[d]$ are defined using the standard abbreviations, while for the greatest fixpoint operator we may put

$$\nu x.\varphi := \neg\mu x.\neg\varphi(\neg x).$$

Note the *triple* use of the negation symbol that is required to maintain the positivity of x — explained by the earlier remarks. ◁

Earlier on we defined the notions of *clean* and *guarded* formulas.

▶ ...

Proposition 3.15 *Every fixpoint formula is equivalent to a clean one.*

Proof. We leave this proof as an exercise for the reader. QED

Proposition 3.16 *Every fixpoint formula is equivalent to a guarded one.*

Proof.(Sketch) We prove this proposition by formula induction. Clearly the only non-trivial case to consider concerns the fixpoint operators. Consider a formula of the form $\eta x.\delta(x)$, where $\delta(x)$ is guarded and clean, and suppose that x has an unguarded occurrence in δ.

First consider an unguarded occurrence of x in $\delta(x)$ inside a fixpoint subformula, say, of the form $\theta y.\gamma(x,y)$. By induction hypothesis, all occurrences of y in $\gamma(x,y)$ are guarded. Obtain the formula $\overline{\delta}$ from δ by replacing the subformula $\theta y.\gamma(x,y)$ with $\gamma(x,\theta y.\gamma(x,y))$. Then clearly $\overline{\delta}$ is equivalent to δ, and all of the unguarded occurrences of x in $\overline{\delta}$ are outside of the scope of the fixpoint operator θ.

Continuing like this we obtain a formula $\eta x.\overline{\delta}(x)$ which is equivalent to $\eta x.\delta(x)$, and in which none of the unguarded occurrences of x lies inside the scope of a fixpoint operator. That leaves $\wedge$ and $\vee$ as the only operation symbols in the scope of which we may find unguarded occurrences of x.

From now on we only consider the case that $\eta = \mu$, the case where $\eta = \nu$ is very similar. Clearly, using the laws of classical propositional logic, we may bring the formula $\overline{\delta}$ into conjunctive normal form

$$(x \vee \alpha_1(x)) \wedge \cdots \wedge (x \vee \alpha_1(x)) \wedge \beta(x), \tag{9}$$

where all occurrences of x in $\alpha_1, \ldots, \alpha_n$ and β are guarded. (Note that we may have $\beta = \top$, or $\alpha_i = \bot$ for some i.)

Clearly (9) is equivalent to the formula

$$\delta'(x) := (x \vee \alpha(x)) \wedge \beta(x),$$

where $\alpha = \alpha_1 \wedge \cdots \wedge \alpha_n$. Thus we are done if we can show that

$$\mu x.\delta'(x) \;\equiv\; \mu x.\alpha(x) \wedge \beta(x). \tag{10}$$

Since $\alpha \wedge \beta$ implies δ', it is easy to see (and left for the reader to prove) that $\mu x.\alpha \wedge \beta$ implies $\mu x.\delta'$. For the converse, it suffices to show that $\varphi := \mu x.\alpha(x) \wedge \beta(x)$ is a prefixpoint of $\delta'(x)$. But it is not hard to derive from $\varphi \equiv \alpha(\varphi) \wedge \beta(\varphi)$ that

$$\delta'(\varphi) = (\varphi \vee \alpha(\varphi)) \wedge \beta(\varphi) \equiv ((\alpha(\varphi) \wedge \beta(\varphi)) \vee \alpha(\varphi)) \wedge \beta(\varphi) \equiv \alpha(\varphi) \wedge \beta(\varphi) \equiv \varphi,$$

which shows that in fact, φ is a fixpoint, and hence certainly a prefixpoint, of $\delta'(x)$. QED

Combining the proofs of the previous two propositions one easily shows the following.

Proposition 3.17 *Every fixpoint formula is equivalent to a clean, guarded one.*

3.3 Game semantics

The formal definition of the semantics of the modal μ-calculus may be mathematically transparent, it is of little help when it comes to unravelling the actual meaning of individual formulas. In practice, it is much easier to work with the *evaluation games* that we are about to introduce now. Obviously, this framework builds on the game-theoretical semantics for ordinary modal logic as described in subsection 2.2, extending it with features for the fixpoint operators. Nice about the game-theoretic semantics is that it brings out the key idea underlying the semantics of fixpoint formulas so clearly:

ν means unfolding, μ means finite unfolding.

From a theoretical perspective, the importance of the game-theoretical semantics of fixpoint logics lies in the fact that the evaluation games are so-called *parity games*, see Chapter 4 for more details. Parity games have a number of very useful properties. In particular, it can be shown that winning strategies for either player can always assumed to be history free, that is, do not depend on moves made earlier in the match, but only on the current position. As we will see further on, this property is crucial in establishing various results about the modal μ-calculus.

3.3.1 The evaluation game

For a definition of the evaluation game of the modal μ-calculus, fix a formula ξ and an LTS $\mathbb{S}$. Without loss of generality we may assume that ξ is *clean*. Basically, the game $\mathcal{E}(\xi, \mathbb{S})$ for ξ a fixpoint formula is defined in the same way as for plain modal logic formulas.

Definition 3.18 $\mathcal{E}(\xi, \mathbb{S})$ is a board game, with players $\exists$ and $\forall$ moving a token around positions of the form $(\varphi, s) \in \mathit{Sfor}(\xi) \times S$. The rules, determining the admissible moves from a given position, together with the player who is supposed to make this move, are given in Table 3. $\lhd$

Clearly, one difference is that the μ-calculus has new formula constructors: the fixpoint operators. These are dealt with in the most straightforward way possible: in a position of the form $(\eta x.\delta, s)$, the next position is *uniquely* determined as the pair (δ, s). Since neither $\exists$ nor $\forall$ thus has any influence over the next position, a position of this form is assigned to neither of the players.

The crucial difference however lies in the treatment of the *bound variables* of ξ. Previously, and still in the case of free variables, positions of the form (p, φ) would be *final positions* of the game, immediately determining the winner of the match. However, at a position (x, s) with x *bound*, the fixpoint variable x gets *unfolded*; this means that the new position is given as (δ_x, s), where $\eta_x x.\delta_x$ is the unique subformula of ξ where x is bound. Note that for this to be well defined, we need ξ to be clean. The disjointness

of $FV(\xi)$ and $BV(\xi)$ ensures that it is always clear whether a variable is to be unfolded or not, and the fact that bound formulas are bound by unique occurrences of fixpoint operators guarantees that δ_x is uniquely determined. Finally, since in this case the next position is also completely determined by the current one, positions of the form (x, s) with x *bound* are not assigned to one of the players.

Position	Player	Admissible moves
$(\varphi_1 \vee \varphi_2, s)$	$\exists$	$\{(\varphi_1, s), (\varphi_2, s)\}$
$(\varphi_1 \wedge \varphi_2, s)$	$\forall$	$\{(\varphi_1, s), (\varphi_2, s)\}$
$(\langle d\rangle\varphi, s)$	$\exists$	$\{(\varphi, t) \mid t \in \sigma_d(s)\}$
$([d]\varphi, s)$	$\forall$	$\{(\varphi, t) \mid t \in \sigma_d(s)\}$
$(\bot, s)$	$\exists$	$\varnothing$
$(\top, s)$	$\forall$	$\varnothing$
(p, s), with $p \in FV(\xi)$ and $s \in V(p)$	$\forall$	$\varnothing$
(p, s), with $p \in FV(\xi)$ and $s \notin V(p)$	$\exists$	$\varnothing$
$(\neg p, s)$, with $p \in FV(\xi)$ and $s \in V(p)$	$\exists$	$\varnothing$
$(\neg p, s)$, with $p \in FV(\xi)$ and $s \notin V(p)$	$\forall$	$\varnothing$
$(\eta_x x.\delta_x, s)$	$-$	$\{(\delta_x, s)\}$
(x, s), with $x \in BV(\xi)$	$-$	$\{(\delta_x, s)\}$

Table 3: Evaluation game for modal fixpoint logic

Example 3.19 Let $\mathbb{S} = \langle S, R, V\rangle$ be the Kripke model with $S = \{0, 1, 2\}$, $R = \{(0, 1), (1, 1), (2, 2)\}$, while V is given by $V(p) = \{2\}$. Now let ξ be the formula $\eta x.p \vee \Box x$, and consider the game $\mathcal{E}(\xi, \mathbb{S})$ initialized at $(\xi, 0)$.

The second position of any match of this game will be $(p \vee \Box x, 0)$ belonging to $\exists$. Clearly, she chooses the disjunct $\Box x$ since otherwise p being false at 0 would mean an immediate loss for her. Now the position $(\Box x, 0)$ belongs to $\forall$ and he will make the only move allowed to him, choosing $(x, 1)$ as the next position. Here an automatic move is made, *unfolding* the variable x, and thus changing the position to $(p \vee \Box x, 1)$. And as before, $\exists$ will choose the right disjunct: $(\Box x, 1)$.

At $(\Box x, 1)$, $\forall$ does have a choice. However it is not difficult to show that choosing $(x, 2)$ would mean that $\exists$ wins the match since p being true at 2 enables her to finally choose the first disjunct of the formula $p \vee \Box x$. So $\forall$ chooses $(x, 1)$, a position that the match already crossed before.

This means that these strategies force the match to be *infinite*, with the variable x unfolding infinitely often at positions of the form $(x, 1)$, and the match taking the following form::

$$(\xi, 0)(p \vee \Box x, 0)(\Box x, 0)(x, 1)(p \vee \Box x, 1)(\Box x, 1)(x, 1)(p \vee \Box x, 1) \ldots$$

So who is declared to be the winner of this match? This is where the difference between the two fixpoint operators shows up. In case $\eta = \mu$, the above infinite match is *lost* by $\exists$ since the fixpoint variable that is unfolded infinitely often is a μ-variable, and μ-variables are to be unfolded only finitely often. In case $\eta = \nu$, the variable unfolded infinitely often is a ν-variable, and this is unproblematic: $\exists$ wins the match. $\lhd$

The above example shows the principle of unfolding at work. Its effect is that matches may now be of infinite length since formulas are no longer deconstructed at every move of the game. Nevertheless, as we will see, it will still be very useful to declare a *winner* of such an infinite game. Implementing the slogan at the beginning of this section, in case of a unique variable that is unfolded infinitely often during a match π, we will declare $\exists$ to be the winner of π if this variable is a ν-variable, and $\forall$ in case we are dealing with a μ-variable. So what happens in case that various variables are unfolded infinitely often? As we will see, in such a case there is always a *unique* such variable that ranks higher than any other.

Definition 3.20 Let ξ be a μPML-formula, and $\mathbb{S}$ a labelled transition system. A *match* of the game $\mathcal{E}(\xi, \mathbb{S})$ is a (finite or infinite) sequence of positions

$$(s, \xi) = (s_0, \varphi_0)(s_1, \varphi_1)(s_2, \varphi_2) \dots$$

which are in accordance with the rules of Table 3. A *full match* is either an infinite match, or a finite match in which the player responsible for the last position got stuck. In practice we will always refer to full matches simply as *matches*. A match that is not full is called *partial*.

Given an infinite match π, we let $Unf^{\infty}(\pi) \subseteq BV(\xi)$ denote the set of variables that are unfolded infinitely often during π. $\lhd$

Proposition 3.21 *Let ξ be a μPML-formula, and $\mathbb{S}$ a labelled transition system. Then for any infinite match π of the game $\mathcal{E}(\xi, \mathbb{S})$, the set $Unf^{\infty}(\pi)$ has a highest ranking member.*

Proof. Since ξ consists of finitely many symbols, $Unf^{\infty}(\pi)$ is not empty. We claim that it is in fact *directed* (with respect to the ranking order). That is, for any x and y in $Unf^{\infty}(\pi)$ there is a variable $z \in Unf^{\infty}(\pi)$ such that $x \leq_{\xi} z$ and $y \leq_{\xi} z$.

For suppose otherwise. Then in particular, $\varphi_x = \eta_x x.\delta_x$ and $\varphi_y = \eta_y y.\delta_y$ are not subformulas of one another. However, π goes through both φ_x and φ_y infinitely often. Now the only way it can move from φ_x to φ_y is by unfolding some variable z such that both φ_x and φ_y are subformulas of φ_z, that is, $x \leq_{\xi} z$ and $y \leq_{\xi} z$. Since this happens infinitely often, some such z must belong to $Unf^{\infty}(\pi)$, as required.

But if $Unf^{\infty}(\pi)$ is directed, being finite it must have a maximum. That is, there is indeed a highest variable in $BV(\xi)$ that gets unfolded infinitely often during π. QED

Given this result, there is now a natural formulation of the winning conditions for infinite matches of evaluation games.

Definition 3.22 The winning conditions of the game $\mathcal{E}(\xi, \mathbb{S})$ are given in Table 4. $\lhd$

	$\exists$ wins π	$\forall$ wins π
π is finite	$\forall$ got stuck	$\exists$ got stuck
π is infinite	$\max(Unf^{\infty}(\pi))$ is a ν-variable	$\max(Unf^{\infty}(\pi))$ is a μ-variable

Table 4: Winning conditions of $\mathcal{E}(\xi, \mathbb{S})$

The point of these evaluation games is that they provide an alternative *yet equivalent* perspective on the semantics of fixpoint formulas. That is, we can prove that for any labelled transition system $\mathbb{S}$, any state s in $\mathbb{S}$, and any clean μPML-formula ξ, we have that

$$\mathbb{S}, s \Vdash \xi \iff (\xi, s) \in \mathrm{Win}(\mathcal{E}(\xi, \mathbb{S})),$$

where $\mathrm{Win}_{\exists}(\mathcal{E}(\xi, \mathbb{S})$ denotes the set of winning positions for $\exists$ in $\mathcal{E}(\xi, \mathbb{S})$. This *adequacy* result is formulated as Theorem 3.26 and will be proved in subsection 3.4.

3.3.2 Examples

Example 3.23 As a first example, consider the formulas $\eta x.p \vee x$. Observe that any match of such a game starts with the positions $(\eta x.p \vee x, s)(p \vee x)$, after which $\exists$ can make a choice. We claim that

$$\mathbb{S}, s \Vdash \mu x.p \vee x \text{ iff } s \in V(p).$$

For the direction from right to left, assume that $s \in V(p)$. Now, if $\exists$ chooses the disjunct p at the position $(s, p \vee x)$, she wins the match because $\forall$ will get stuck at (s, p). Hence $s \in \mathrm{Win}_{\exists}(\mathcal{E}(\eta x.p \vee x, \mathbb{S}))$.

On the other hand, if $s \notin V(p)$, then $\exists$ will lose if she chooses disjunct p at position $(s, p \vee x)$. So she must choose the disjunct x which then unfolds to $p \vee x$ so that $\exists$ is back at the position $(s, p \vee x)$. Thus if $\exists$ does not want to get stuck her only way to survive is to keep playing the position (s, x), thus causing the match to be infinite. But such a match is won by $\forall$ since the only variable that gets unfolded infinitely often is a μ-variable. So in this case we see that $s \notin \mathrm{Win}_{\exists}(\mathcal{E}(\eta x.p \vee x, \mathbb{S}))$.

If on the other hand we take $\eta = \nu$, then $\exists$ can win any match:

$$\mathbb{S}, s \Vdash \nu x.p \vee x.$$

It is easy to see that the strategy of always choosing the disjunct x at a position of the form $(s, p \vee x)$ is winning. For, it forces all games to be infinite, and since the only fixpoint variable that gets ever unfolded here is a ν-variable, all infinite matches are won by $\exists$. $\lhd$

Example 3.24 Now we turn to the formulas $\mu x.\Diamond x$ and $\nu x.\Diamond x$. First consider how a match for any of these formulas proceeds. The first two positions of such a match will be of the form $(\eta x.\Diamond x, s)(\Diamond x, s)$, at which point it is $\exists$'s turn to make a move. Now she either is stuck (in case the state s has no successor) or else the next two positions are $(x, t)(\Diamond x, t)$ for some successor t of s, chosen by $\exists$. Continuing this analysis, we see that there are two possibilities for a match of the game $\mathcal{E}(\eta x.\Diamond x, \mathbb{S})$:

1. the match is an infinite sequence of positions

   $$(\eta x.\Diamond x, s_0)(\Diamond x, s_0)(x, s_1)(\Diamond x, s_1)(x, s_2)\dots$$

   corresponding to an infinite path $s_0 R s_1 R s_2 R \dots$ through $\mathbb{S}$.

2. the match is an finite sequence of positions

   $$(\eta x.\Diamond x, s_0)(\Diamond x, s_0)(x, s_1)(\Diamond x, s_1)\dots(\Diamond x, s_k)$$

   corresponding to a finite path $s_0 R s_1 R \dots s_k$ through $\mathbb{S}$, where s_k has no successors.

Note too that in either case it is only $\exists$ who has turns, and that her strategy corresponds to choosing a *path* through $\mathbb{S}$. From this it is easy to derive that

$$\begin{array}{ll} \mathbb{S}, s \not\Vdash \mu x.\Diamond x, & \\ \mathbb{S}, s \Vdash \nu x.\Diamond x & \text{iff} \quad \text{there is an infinite path starting at } s. \end{array}$$

$\lhd$

▶ `Until operator`

Until now all examples that we consider involved only a single fixpoint operator. Let us now consider a slightly more involved example, containing both a least and a greatest fixpoint operator.

Example 3.25 Let ξ be the following formula:

$$\xi = \nu x.\mu y.\underbrace{(p \wedge \Diamond x)}_{\alpha_p} \vee \underbrace{(\neg p \wedge \Diamond y)}_{\alpha_{\neg p}}$$

Then we claim that for any LTS $\mathbb{S}$, and any state s in $\mathbb{S}$:

$$\mathbb{S}, s \Vdash \xi \text{ iff } \text{ there is some path from } s \text{ on which } p \text{ is true infinitely often.} \qquad (11)$$

To see why this is so, first suppose that there is a path $\pi = s_0 s_1 s_2 \dots$ as described in the right hand side of (11). Now suppose that $\exists$ plays according to the following strategy:

(a) at a position $(\alpha_p \vee \alpha_{\neg p}, t)$, choose (α_p, t) if $\mathbb{S}, t \Vdash p$ and choose $(\alpha_{\neg p}, t)$ otherwise;

(b) at a position $(\Diamond\varphi, t)$, choose (φ, s_{k+1}) if $t = s_k$ on the path, and choose an arbitrary successor (if possible) if t is off the path.

We claim that this is a winning strategy for $\exists$ in the evaluation game initialized at (ξ, s). For consider a match of this game. Since $\exists$ always chooses the propositionally safe disjunct of $\alpha_p \vee \alpha_{\neg p}$, she forces $\forall$, when faced with a position of the form $(\alpha_{\pm p}, t) = (\pm p \wedge \Diamond z, t)$ to always choose the diamond conjunct $\Diamond z$, or lose immediately. In this way she guarantees that she always gets to positions of the form $(s_i, \Diamond z)$, and thus she can force the match to last infinitely long, following the infinite path π. But why does she actually *win* this match? The point is that whenever she chooses α_p, three moves later, x will be unfolded, and likewise with $\alpha_{\neg p}$ and y. Thus, p being true infinitely often on π means that the ν-variable x gets unfolded infinitely often. And so, even though the μ-variable y may perhaps get unfolded infinitely often as well, she wins the match since x ranks higher than y anyway.

For the other direction, assume that $\mathbb{S}, s \Vdash \xi$ so that $\exists$ has a winning strategy in the game $\mathcal{E}(\xi, \mathbb{S})$ initialized at (ξ, s). It should be clear that any winning strategy must follow (a) above. So whenever $\forall$ faces a position $(p \wedge \Diamond z, t)$, p will be true, and likewise with positions $(\neg p \wedge \Diamond z, t)$. Now consider a match in which $\forall$ plays propositionally sound, that is, always chooses the diamond conjunct of these positions. This match must be infinite since neither of the players will ever get stuck: $\forall$ not, because he can always choose a diamond conjunct, and $\exists$ not, because we assumed her strategy to be winning. But a second consequence of $\exists$'s strategy being winning, is that it cannot happen that y is unfolded infinitely often, while x is not. So x is unfolded infinitely often, and as before, x only gets unfolded right after the match passed a world where p is true. Thus the path chosen by $\exists$ must contain infinitely many states where p holds. $\lhd$

3.4 Adequacy

In this section we prove the *adequacy* of the game semantics: We will show that a fixpoint formula ξ is true at a state precisely if $\exists$ has a winning strategy in the corresponding initialized evaluation game.

Theorem 3.26 *Let ξ be a clean μPML-formula in positive format. Then for all labelled transition systems $\mathbb{S}$ and all states s in $\mathbb{S}$:*

$$\mathbb{S}, s \Vdash \xi \iff (\xi, s) \in \mathrm{Win}(\mathcal{E}(\xi, \mathbb{S})).$$

Proof. The theorem is proved by induction on the complexity of ξ. We only discuss the case that ξ is of the form $\mu x.\delta$, leaving the other cases as exercises to the reader.

$\boxed{\Rightarrow}$ For the direction from left to right, it suffices to show that the set $W := \{s \in S \mid (\xi, s) \in \mathrm{Win}(\mathcal{E}(\xi, \mathbb{S}))\}$ is a *prefixpoint* of δ in $\mathbb{S}$, for this implies that $[\![\xi]\!]^{\mathbb{S}} \subseteq W$ by definition of the semantics of the least fixpoint operator. Showing that $W \in \mathrm{PRE}(\delta_x^{\mathbb{S}})$ boils down to proving that $\mathbb{S}[x \mapsto W] \Vdash \delta \to x$. Abbreviate $\mathbb{S}' := \mathbb{S}[x \mapsto W]$, and assume that $\mathbb{S}', t \Vdash \delta$. We will prove that $\mathbb{S}', t \Vdash x$, or equivalently, that $t \in W$. In other words, we have to provide $\exists$ with a winning strategy in the game $\mathcal{G}@(\xi, t)$.

To start with, it inductively follows from $\mathbb{S}', t \Vdash \delta$ that $\exists$ has a winning strategy f' in the game $\mathcal{G}' := \mathcal{E}(\delta, \mathbb{S}')$ initialized at (δ, t). But $\mathcal{G}@(\xi, t)$ and $\mathcal{G}'@(\delta, t)$ are *very* similar games: apart from the starting position (ξ, t) of the first game — which will be immediately replaced with the initial position (δ, t) of the second game anyway — the positions of the two games are exactly the same. In fact, the only real difference between the games shows up in the rule concerning positions of the form (x, u). In $\mathcal{G}'$, x is a *free* variable ($x \in FV(\delta)$), so in a position (u, x) the game is over, the winner being determined by whether $u \in W$ or not. In $\mathcal{G}$ however, x is *bound*, so at a state (u, x), the variable x will get unfolded.

Second, observe that by definition of W, for every state $w \in W$, $\exists$ has a winning strategy f_w for the variant of game $\mathcal{G}$ starting at (ξ, w). Clearly this strategy is also winning in the same game $\mathcal{G}$ if we take (x, w) as starting position: In both variants of $\mathcal{G}$, the second position will be the pair (δ, w).

Now consider the following strategy g for $\exists$ in $\mathcal{G}$:

- after the initial move, the position of the match is (δ, t);
- $\exists$ first plays her strategy f' (note that this is well-defined);
- as soon as a position (u, x) is reached, distinguish cases:
  - if $u \in W$ then $\exists$ continues with f_u;
  - if $u \notin W$ then $\exists$ continues with a random strategy.

We claim that this g is a winning strategy for $\exists$ in $\mathcal{G}$. To see this, make the following case distinction concerning an arbitrary match π which is consistent with g:

No state (u, x) is reached. This means that π, seen as a $\mathcal{G}'$-match, is won by $\exists$. Since $\mathcal{G}$ and $\mathcal{G}'$ only differ when it comes to x this means that π is also a win for $\exists$ in $\mathcal{G}$. Note that here it does not matter whether π is finite or infinite.

At some stage a position (u, x) is reached. In the $\mathcal{G}'$-perspective on π, the match would have reached a final position. Since f' was a winning strategy for $\exists$, this can only happen if $u \in W$. (In other words, the second case mentioned above does not occur.) So $\exists$ consequently plays according to f_u; note that the first position after (u, x) is (u, δ). But by definition, f_u is a winning strategy for $\exists$ in $\mathcal{G}@(u, \delta)$. It is then easy to see that any continuation of the match in which $\exists$ plays f_u, is won by $\exists$.

Altogether this shows that indeed, g is a winning strategy for $\exists$.

$\boxed{\Rightarrow}$ For the opposite direction, assume that $\exists$ has a winning strategy f for the evaluation game $\mathcal{E}(\xi, \mathbb{S})$ initialized at (ξ, s). Suppose for contradiction that $s \notin [\![\xi]\!]^{\mathbb{S}}$.

First consider an arbitrary point $t \notin [\![\xi]\!]^{\mathbb{S}}$. Since $[\![\xi]\!]^{\mathbb{S}}$ is a prefixpoint of δ, we have $\mathbb{S}' \Vdash \delta \to x$ where $\mathbb{S}' := \mathbb{S}[x \mapsto [\![\xi]\!]^{\mathbb{S}}]$. Then from $\mathbb{S}', s \nVdash x$ (because $s \notin [\![\xi]\!]^{\mathbb{S}}$) it follows that $\mathbb{S}', t \nVdash \delta$. Thus by the inductive hypothesis it follows that $\exists$ does not have a winning strategy for the game $\mathcal{G}' = \mathcal{G}(\delta, \mathbb{S}')@(\delta, t)$. That is, for each strategy g of $\exists$ starting at (δ, t), $\forall$ has a counter strategy $\bar{g}_t$ such that the match of $\mathcal{G}'$ determined by g and $\bar{g}_t$ is won by $\forall$. Furthermore, note that by the resemblance of the games $\mathcal{G}$ and $\mathcal{G}'$, the strategy f may be taken as a strategy in $\mathcal{G}'@(\delta, s)$ as well.

Now consider the matches of $\mathcal{G}$, starting at (ξ, s), in which $\exists$ plays according to her supposedly winning strategy f. Suppose that $\forall$ counters the strategy f as follows:

- $\forall$ starts with the strategy $\bar{f}_s$;
- from that moment on, $\forall$ sticks to his current strategy, unless a position (x, u) is reached; now distinguish cases:
  1. if $u \in Q$ then $\forall$ continues with a random strategy;
  2. if $u \notin Q$ then $\forall$ plays as follows. Let β be the match this far (including (x, u)), and let f_β denote the strategy of $\exists$ for the $\mathcal{G}$-game starting at (δ, u) given by $f_\beta(\gamma) = f(\beta\gamma)$. Then by our earlier discussion, f_β can be seen as an $\mathcal{G}'$-strategy for matches starting at (δ, u), and so $\forall$ may adopt his counter strategy $(\bar{f}_\beta)_u$ from this moment on.

Consider the $\mathcal{G}$-match β starting at $(\mu x.\delta, s)$ determined by $\exists$ playing her strategy f and $\forall$ using the strategy defined above. First observe that β can pass through positions of the form (x, u) only finitely many times, for otherwise, the μ-variable x would be the highest fixed point variable unfolded infinitely often, contradicting the assumption that f is winning for $\exists$. Second, observe that the first possibility of passing x-positions mentioned above will never occur. This is because arriving at a position (x, u) with $u \in Q$ would mean that, contrary to our earlier conclusion, $\exists$ would have a successful strategy in $\mathcal{G}'$ at a point $v \notin Q$ after all.

This means, however, that after a certain initial partial play β, ending in a position (x, u) with $u \notin Q$, $\forall$ will stick to his strategy $(\bar{f}_\beta)_u$, while no further position (v, x) is ever reached. It follows from our assumptions on $(\bar{f}_\beta)_u$ that the match γ resulting from $\exists$ playing f_β against $\forall$ playing $(\bar{f}_\beta)_u$ is winning for $\forall$ in $\mathcal{G}'$, and from this it is not hard to derive that the $\mathcal{G}$-match $\delta = \beta\gamma$ is won by $\forall$. This provides the desired contradiction, since it shows that the strategy f is not winning for $\exists$ after all. QED

3.5 Coalgebraic modal fixpoint logic

In section 2.6 we introduced a coalgebraic reformulation of the syntax and semantics of modal logic, introducing the bullet modality. It is in fact straightforward to give a similar coalgebraic version of modal *fixpoint* logic.

Definition 3.27 Formulas of the language $\mu\mathrm{CML}_{\mathsf{D}}(\mathsf{P})$ are given by the following recursive definition:

$$\varphi \ ::= \ x \mid \bot \mid \top \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \pi \bullet \Phi \mid \mu x.\varphi \mid \nu x.\varphi$$

where π denotes a subset of P, and Φ is a D-indexed set of $\mu\mathrm{CML}_{\mathsf{D}}(\mathsf{P})$-formulas. $\lhd$

An interesting observation on the plain (i.e. fixpoint-free) version of this language, Theorem 2.28 states that every formula of CML can be rewritten into an equivalent version that only uses the special bullet conjunction. An important result, here formulated as Theorem 9.5, states that the same observation also applies to the language CML introduced in Definition 3.27. In fact, this is perhaps the most fundamental results underlying the theory of the modal μ-calculus, and one of the main reasons for introducing the coalgebraic reformulation of μPML. Before we can prove this theorem, however, quite a bit of work is needed.

4 Board games

Much of the work linking (fixpoint) logic to automata theory involves nontrivial concepts and results from the theory of infinite games. In this chapter we discuss some of the highlights of this theory in a fair amount of detail. This allows us to be rather informal about game-theoretic concepts in the rest of the notes.

4.1 Board games

The games that we are dealing with here can be classified as *board* or *graph games*. They are played by two agents, here to be called 0 and 1.

Definition 4.1 If $P \in \{0,1\}$ is a player, then $\bar{P}$ denotes the *opponent* $1-P$ of P. $\lhd$

A board game is played on a *board*, which is nothing but a directed graph, in which each node is marked with either 0 or 1. A *match* of the game consists of the two players moving a token across the board, following the edges of the graph. To regulate this, the collection of graph nodes, usually referred to as *positions* of the game, is partitioned into two sets, one for each player. Thus with each position we may associate a unique player whose turn it is to move when the token lies on position p.

Definition 4.2 A *board* is a structure $\mathbb{B} = \langle B_0, B_1, E\rangle$, such that B_0 and B_1 are disjunct, and $E \subseteq B^2$, where $B := B_0 \cup B_1$. We will make use of the notation $E[p]$ for the set of *admissible moves* from a board position $p \in B$, that is, $E[p] := \{q \in B \mid (p,q) \in E\}$. Positions not in $E[p]$ will sometimes be referred to as *illegitimate moves* with respect to p. A position $p \in B$ is a *dead end* if $E(p) = \varnothing$. If $p \in B$, we let P_p denote the (unique) player such that $p \in B_{P_p}$, and say that p *belongs to* P_p, or that it is P_p's *turn* to move at p. $\lhd$

A match of the game may in fact be identified with the sequence of positions visited during play, and thus corresponds to a *path* through the graph.

Definition 4.3 A *path* through a board $\mathbb{B} = \langle B_0, B_1, E\rangle$ is a (finite or infinite) sequence $\pi \in B^\infty$ such that $E\pi_i\pi_{i+1}$ whenever applicable. A *match* through $\mathbb{B}$ is either an infinite $\mathbb{B}$-path, or a finite $\mathbb{B}$-path π ending with a dead end (i.e. $E[last(\pi)] = \varnothing$).

A *partial match* is a finite path through $\mathbb{B}$ that is not a match; in other words, the last position of a partial match is not a dead end. We let PM_P denote the set of partial matches such that P is the player whose turn it is to move at the last position of the match. In the sequel, we will denote this player as P_π; that is, $P_\pi := P_{last(\pi)}$. $\lhd$

Each full or completed match is *won* by one of the players, and *lost* by their opponent; that is, there are no draws. A finite match ends if one of the players gets *stuck*,

that is, is forced to move the token from a position without successors. Such a finite, completed, match is lost by the player who got stuck. If neither player ever gets stuck, an infinite match arises. The flavor of a board game is very much determined by the winning conditions of these infinite matches.

Definition 4.4 Given a board $\mathbb{B}$, a *winning condition* is a map $W : B^\omega \to \{0, 1\}$. An infinite match π is *won* by $W(\pi)$. A *board game* is a structure $\mathcal{G} = \langle B_0, B_1, E, W\rangle$ such that $\langle B_0, B_1, E\rangle$ is a board, and W is a winning condition on B. $\lhd$

Although the winning condition given above applies to all infinite B-sequences, it will only make sense when applied to matches. We have chosen the above definition because it is usually much easier to formulate maps that are defined on all sequences.

Before players can actually start playing a game, they need a starting position. The following definition introduces some terminology and notation.

Definition 4.5 An *initialized board game* is a pair consisting of a board game $\mathcal{G}$ and a position q on the board of the game; such a pair is usually denoted $\mathcal{G}@q$.

Given a (partial) match π, its first element $\mathit{first}(\pi)$ is called the *starting position* of the match. We let $\mathrm{PM}_P(q)$ denote the set of partial matches for P that start at position q. $\lhd$

Central in the theory of games is the notion of a *strategy*. Roughly, a strategy for a player is a set of instructions advising the player how to continue partial matches when it is their turn to move. More precisely, a strategy maps partial plays for the player to new positions, with the proviso that the new position must be a legitimate continuation of the partial match.

Definition 4.6 Given a board game $\mathcal{G} = \langle B_0, B_1, E, W\rangle$ and a player P, a *P-strategy*, or a *strategy for P*, is a map $f : \mathrm{PM}_P \to B$ such that $E(\mathit{last}(\pi), f(\pi))$. A strategy for the game $\mathcal{G}@q$ is a map $f : \mathrm{PM}_P(q) \to B$ satisfying this condition.

A match π is *consistent* with a P-strategy f if for any $\pi' \sqsubset \pi$ with $\mathit{last}(\pi') \in B_P$, the next position on π is indeed the element $f(\pi)$.

A P-strategy f is *winning for P* if P wins every play that is consistent with f. A position $q \in B$ is *winning for P* if P has a winning strategy for the game $\mathcal{G}$ starting at q; the collection of winning positions for P in $\mathcal{G}@q$ is denoted as $\mathrm{Win}(\mathcal{G}@q)$. $\lhd$

Convention 4.7 In practice, when defining strategies, it will often be convenient to extend the definition of a strategy to include maps f that do not necessarily satisfy the condition that $E(\mathit{last}(\pi), f(\pi))$ for every partial play π. In such a case we will say that the map prescribes an *illegitimate* move in the partial play π. We will only permit ourselves such a sloppiness in a context where in fact the partial play π is not consistent with the pseudo-strategy f, and thus the situation where the pseudo-strategy would actually ask for an illegitimate move will not occur.

Definition 4.8 The game $\mathcal{G}$ on the board $\mathbb{B}$ is *determined* if $\mathrm{Win}_0(\mathcal{G}) \cup \mathrm{Win}_1(\mathcal{G}) = B$; that is, each position is winning for one of the players. ◁

In principle, when deciding how to move in a match of a board game, players may use information about the entire history of the match played thus far. However, it will turn out to be advantageous to work with strategies that are simple to compute. This applies for instance to so-called finite memory strategies, which can be computed using only a finite amount of information about the history of the match.

▶ `discuss finite memory strategy`

Particularly nice are so-called *history-free* or *memoryless* strategies, which only depend on the current position (i.e., the final position of the partial play). These will be critically needed in the proofs of some of the most fundamental results in the area of logic and automata theory, such as the Theorems 6.17 and 9.9.

Definition 4.9 A strategy f is *history free* if $f(\pi) = f(\pi')$ for any π, π' with $last(\pi) = last(\pi')$. ◁

4.2 Winning conditions

In case we are dealing with a *finite* board B, then we may nicely formulate winning conditions in terms of the set of positions that occur *infinitely often* in a given match. But in the case of an infinite board, there may be matches in which no position occurs infinitely often (or more than once, for that matter). Nevertheless, we may still define winning conditions in terms of objects that occur infinitely often, if we make use of *finite colorings* of the board. If we assign to each position $b \in B$ a *color*, taken from a finite set C of colors, then we may formulate winning conditions in terms of the *colors* that occur infinitely often in the match.

Definition 4.10 A *coloring* of B is a function Γ assigning to each position $p \in B$ a *color* $\Gamma(b)$ taken from some finite set C of colors. Such a coloring $\Gamma : B \to C$ naturally extends to a map $\Gamma : B^\omega \to C^\omega$ by putting $\Gamma(p_0 p_1 \dots) := \Gamma(p_0)\Gamma(p_1)\dots$. ◁

Now if $\Gamma : B \to C$ is a coloring, for any infinite sequence $\pi \in B^\omega$, the map $\Gamma \circ \pi$ forms the associated sequence of colors. But then since C is finite there must be some elements of C that occur infinitely often in this stream.

Definition 4.11 Let $\mathbb{B}$ be a board and $\Gamma : B \to C$ a coloring of B. Given an infinite sequence $\pi \in B^\omega$, we let $Inf_\Gamma(\pi)$ denote the set of colors that occur infinitely often in the sequence $\Gamma \circ \pi$.

A *Muller condition* is a collection $\mathcal{M} \subseteq \wp(C)$ of subsets of C. The corresponding winning condition is defined as the following map $W_{\mathcal{M}} : B^\omega \to \{0,1\}$:

$$W_{\mathcal{M}}(\pi) := \begin{cases} 0 & \text{if } Inf_\Gamma(\pi) \in \mathcal{M} \\ 1 & \text{otherwise.} \end{cases}$$

A *Muller game* is a board game of which the winning conditions are specified by a Muller condition. ◁

In words, player 0 wins an infinite match $\pi = p_0 p_1 \ldots$ if the set of colors one meets infinitely often on this path, belongs to the Muller collection $\mathcal{M}$.

▶ `Examples to be supplied.`

Muller games have two nice properties. First, they are determined. This follows from a well-known general game-theoretic result, but can also be proved directly. In addition, we may assume that the winning strategies of each player in a Muller game are finite memory strategies.

▶ `Details to be supplied`

The latter property becomes even nicer if the Muller condition allows a formulation in terms of a *parity map*. In this case, as colors we take natural numbers. Note that by definition of a coloring, the range $\Omega[B]$ of the coloring function Ω is finite. This means that every subset of $\Omega[B]$ has a maximal element. Hence, every match determines a unique natural number, namely, the 'maximal color' that one meets infinitely often during the match. Now a parity winning condition states that the winner of an infinite match is 0 if this number is even, and 1 if it is odd. More succinctly, we formulate the following definition.

Definition 4.12 Let B be some set; a *parity map* on B is a coloring $\Omega : B \to \omega$, that is, a map of finite range. A *parity game* is a board game $\mathcal{G} = \langle B_0, B_1, E, W_\Omega \rangle$ in which the winning condition is given by

$$W_\Omega(\pi) \;:=\; \max(Inf_\Omega(\pi)) \mod 2.$$

Such a parity game is usually denoted as $\mathcal{G} = \langle B_0, B_1, E, \Omega \rangle$. ◁

The key property that makes parity games so interesting is the following.

Theorem 4.13 (History-Free Determinacy of Parity Games) *For any parity game $\mathcal{G}$ there are history-free strategies f_0 and f_1 for 0 and 1, respectively, such that for every position q there is a player P such that f_P is a winning strategy for P in $\mathcal{G}@q$.*

▶ `Proof of this theorem to be supplied`

5 Stream automata

As we already mentioned in the introduction, automata are of fundamental importance in the theory of the modal μ-calculus and other fixpoint logics. This chapter gives an introduction to the theory of automata operating on (potentially infinite) objects. In particular, we discuss issues such as determinism, nondeterminism, and alternation, together with the game-theoretic definition of acceptance.

Whereas in the next chapter we will meet various kinds of automata for classifying (pointed) Kripke structures, here we confine our attention to the devices that operate on *streams* or infinite words, these being the simplest nontrivial examples of infinite behavior.

Definition 5.1 Given an alphabet C, a *C-stream* or *infinite word over C* is just an infinite C-sequence. Sets of C-streams are called *ω-languages* over C. $\lhd$

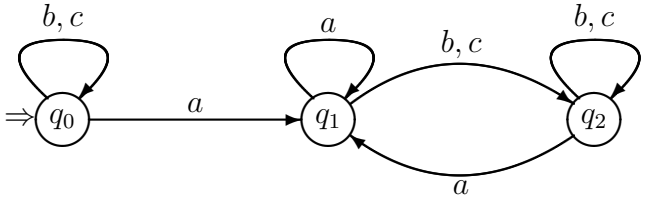
The material treated in this chapter may be completely standard, our perspective is less so. In particular, at a certain moment we will make a switch from looking at streams themselves to considering objects that *produce* such streams. These objects will be called *stream coalgebras*, and we believe that our *coalgebraic* point of view has some conceptual advantages, which may perhaps become fully clear in the next chapter.

5.1 Deterministic stream automata

We start with the standard definition.

Definition 5.2 Given an *alphabet* C, a *deterministic C-automaton* is a quadruple $\mathbb{A} = \langle A, \delta, Acc, a_I \rangle$, where A is a finite set, $a_I \in A$ is the *initial state* of $\mathbb{A}$, $\delta : A \times C \to A$ its *transition function* of $\mathbb{A}$, and $Acc \subseteq A^\omega$ its *acceptance condition*. The pair $\langle A, \delta \rangle$ is called the *transition diagram* of $\mathbb{A}$. $\lhd$

Example 5.3 The transition diagram and initial state of a deterministic automaton can nicely be represented graphically, as in the picture below, where $C = \{a, b, c\}$:



An automaton comes to live if we supply it with input, in the form of a stream over its alphabet: It will *process* this stream, as follows. Starting from the initial state a_I, the automaton will step by step pass through the stream, jumping from one state to another as prescribed by the transition function.

For example: let $\mathbb{A}_0$ be any automaton with transition diagram and initial state as given above, and suppose that we give this device as input the stream $\alpha = abcabcabcabcabc\cdots$. Then we find that $\mathbb{A}_0$ will make an infinite series of transitions, determined by α:

$$q_0 \stackrel{a}{\to} q_1 \stackrel{b}{\to} q_2 \stackrel{c}{\to} q_2 \stackrel{a}{\to} q_1 \cdots$$

Thus the machine passes through an infinite sequence of states:

$$\rho = q_0q_1q_2q_2q_1q_2q_2q_1q_2q_2\ldots$$

This sequence is called the *run* of the automaton on the word α — a run of $\mathbb{A}$ is thus an A-stream.

For a second example, on the word $\alpha' = abacabcbcbcbcbcb\cdots$ the run of the automaton $\mathbb{A}_0$ looks as follows:

$$q_0 \stackrel{a}{\to} q_1 \stackrel{b}{\to} q_2 \stackrel{a}{\to} q_1 \stackrel{c}{\to} q_2 \stackrel{a}{\to} q_1 \stackrel{b}{\to} q_2 \stackrel{c}{\to} q_2 \stackrel{b}{\to} q_2 \stackrel{c}{\to} \cdots$$

we see that from the sixth step onwards, the machine device remains circling in its state q_2: $\cdots\ q_2 \stackrel{b}{\to} q_2 \stackrel{c}{\to} q_2 \stackrel{b}{\to} \cdots$. ◁

Definition 5.4 The *run* of a finite automaton $\mathbb{A} = \langle A, \delta, Acc, a_I\rangle$ on an C-stream $\gamma = c_0c_1c_2\ldots$ is the infinite A-sequence

$$\rho = a_0a_1a_2\ldots$$

such that $a_0 = a_I$ and $a_{i+1} = \delta(a_i, c_i)$ for every $i \in \omega$. ◁

In order to determine which runs are successful, we need the acceptance condition.

Definition 5.5 A run $\rho \in A^\omega$ is *successful* with respect to an acceptance condition Acc if $\rho \in Acc$.

A finite C-automaton $\mathbb{A} = \langle A, \delta, Acc, a_I\rangle$ *accepts* a C-stream γ if the (unique) run of $\mathbb{A}$ on γ is successful. The *ω-language* $L_\omega(\mathbb{A})$ associated with $\mathbb{A}$ is defined as the set of streams that are accepted by $\mathbb{A}$. Two automata are called *equivalent* if they accept the same streams. ◁

Very often, the acceptance condition is defined in terms of the states of the automaton that are visited infinitely often during a run.

Definition 5.6 Given an infinite sequence α over some finite set A, let $Occ(\alpha)$ and $Inf(\alpha)$ denote the set of elements of A that occur in α at least once, respectively, infinitely often. ◁

Definition 5.7 Given a transition diagram $\langle A, \delta \rangle$, we define the following types of acceptance conditions:

- A *Muller* condition is given as a collection $\mathcal{M} \subseteq \wp(A)$ of subsets of A. The corresponding acceptance condition is defined as

$$Acc_{\mathcal{M}} := \{\alpha \in A^\omega \mid Inf(\alpha) \in \mathcal{M}\}.$$

- A *Büchi* condition is given as a subset $F \subseteq A$. The corresponding acceptance condition is defined as

$$Acc_F := \{\alpha \in A^\omega \mid Inf(\alpha) \cap F \neq \varnothing\}.$$

- A *parity condition* is given as a map $\Omega : A \to \omega$. The corresponding acceptance condition is defined as

$$Acc_\Omega := \{\alpha \in A^\omega \mid \max(Inf(\Omega \circ \alpha)) \text{ is even } \}.$$

Automata with these acceptance conditions are called *Muller*, *Büchi* and *parity automata*, respectively. ◁

Of these three types of acceptance conditions, the Muller condition perhaps is the most natural. It exactly and directly specifies the subsets of A that are admissible as the set $Inf(\rho)$ of a successful run. The Büchi condition is also fairly intuitive: an automaton with Büchi condition F accepts a stream α if the run on α infinitely often passes through some state in F. This makes Büchi automata the natural analog of the automata that operate on *finite* words.

▶ `elaborate`

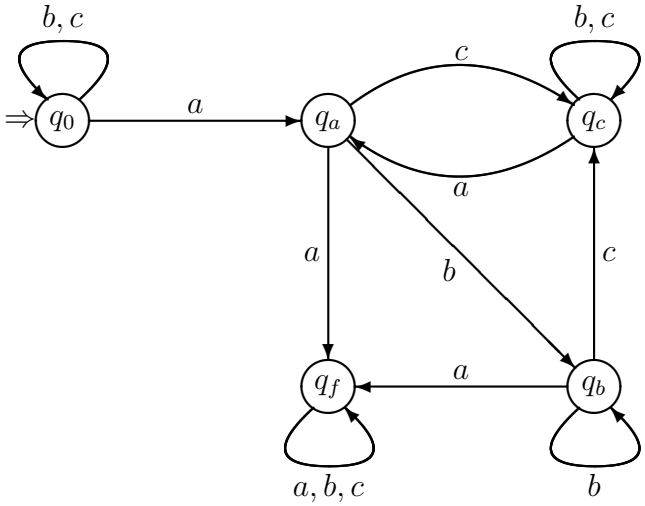
The parity condition at first sight seems rather artificial. Nevertheless, for a number of reasons the parity automaton is destined to play the leading role in these notes. Most importantly, the distinction between even and odd parities directly corresponds to that between least and greatest fixpoint operators, so that parity automata are the more direct automata-theoretic counterparts of fixpoint formulas. An additional theoretic motivation to use parity automata is that their associated acceptance games have some very nice game-theoretical properties, see Theorem 4.13.

Example 5.8 Suppose that we supply the device of Example 5.3 with the Büchi acceptance condition $F_0 = \{q_1\}$. That is, the resulting automaton $\mathbb{A}_0$ accepts a stream α iff the run of $\mathbb{A}_0$ passes through the state q_1 infinitely often. For instance, $\mathbb{A}_0$ will accept the word $\alpha = abcabcabcabcabc\cdots$, because the run of $\mathbb{A}_0$ is the stream $q_0q_1q_2q_2q_1q_2q_2q_1q_2q_2\ldots$ which indeed contains q_1 infinitely many times. On the other

hand, as we saw already, the run of $\mathbb{A}_0$ on the stream $\alpha' = abacabcbcbcbcbcb\cdots$ loops in state q_2, and so α' will not be accepted.

In general, it is not hard to prove that $\mathbb{A}_0$ accepts a C-stream γ iff γ contains infinitely many a's. $\lhd$

Example 5.9 Consider the automaton $\mathbb{A}_1$ given by the following diagram and initial state:



The Muller acceptance condition of the automaton is given as the set

$$\{\ \{q_0\}\ ,\ \{q_c\}\ ,\ \{q_a, q_c\}\ ,\ \{q_a, q_b, q_c\}\ \}$$

We leave it as an exercise for the reader that this automaton accepts those infinite streams in which every a is followed by a finite number of b's, followed by a c. $\lhd$

▶ `Example of parity automaton to be added`

It is important to understand the relative strength of Muller, Büchi and parity automata when it comes to recognizing ω-languages. The Muller acceptance condition is the more fundamental one in the sense that the other two are easily represented by it.

Proposition 5.10 *There is an effective procedure transforming a deterministic Büchi stream automaton into an equivalent deterministic Muller stream automaton.*

Proof. Given a Büchi condition F on a set A, define the corresponding Muller condition $\mathcal{M}_F \subseteq \wp(A)$ as follows:

$$\mathcal{M}_F := \{B \subseteq A \mid B \cap F \neq \varnothing\}.$$

Trivially then, $Acc_{\mathcal{M}_F} = Acc_F$. It is now immediate that any Büchi automaton $\mathbb{A} = \langle A, \delta, F, a_I\rangle$ is equivalent to the Muller automaton $\langle A, \delta, \mathcal{M}_F, a_I\rangle$. QED

Proposition 5.11 *There is an effective procedure transforming a deterministic parity stream automaton into an equivalent deterministic Muller stream automaton.*

Proof. Analogous to the proof of the previous proposition. QED

Interestingly enough, Muller automata can be simulated by devices with a parity condition.

Proposition 5.12 *There is an effective procedure transforming a deterministic Muller stream automaton into an equivalent deterministic parity stream automaton.*

Proof. Given a Muller automaton $\mathbb{A} = \langle A, \delta, \mathcal{M}, a_I \rangle$, define the corresponding parity automaton $\mathbb{A}' = \langle A', \delta', \Omega, a'_I \rangle$ as follows. The crucial concept used in this construction is that of *latest appearance records*. The following notation will be convenient: given a sequence in A^*, say, $\alpha = a_1 \dots a_n$, we let $\widetilde{\alpha}$ denote the set $\{a_1, \dots, a_n\}$, and $\alpha[\nabla/a]$ the sequence α with every occurrence of a being replaced with the symbol ∇.

To start with, the set A' of states is defined as the collection of those finite sequences over the set $A \cup \{\nabla\}$ in which every symbol occurs exactly once:

$$A' = \{a_1 \dots a_k \nabla a_{k+1} \dots a_m \mid m = |A| \text{ and } A = \{a_1, \dots, a_m\}\}.$$

Intuitively, these states encode information about the states of $\mathbb{A}$ that have been visited during some initial part of its run on some word. Given a state $\alpha\nabla\beta$ with $\alpha = a_1 \dots a_k$ and $\beta = a_{k+1} \dots a_m$, this information consists of two parts. First, there is an n such that the states $a_1 \dots a_n$ are the states that have *never* been visited during the run, and the remaining sequence $a_{n+1} \dots a_m$ lists the states that have been visited, in reverse order. Second then, the ∇ marks the *previous position* of a_m in the list. Note that it follows from this that $s \leq k$.

For the initial position of $\mathbb{A}'$, let $a_1, \dots, a_m$ be some enumeration of A with $a_I = a_m$, and define

$$a'_I \;:=\; a_1 \dots a_m \nabla.$$

For the transition function, consider a state $\alpha = a_1 \dots a_k \nabla a_{k+1} \dots a_m$ in A', and a color $c \in C$. To obtain the state $\delta'(\alpha, c)$, replace the occurrence of $\delta(a_m, c)$ in $a_1 \dots a_m$ with ∇, and place state $\delta(a_m, c)$ itself right after the result of this substitution. Thus the ∇ in the new sequence marks the latest appearance of the state $\delta(a_m, c)$. Formally, we put

$$\delta'(a_1 \dots a_k \nabla a_{k+1} \dots a_m, c) \;:=\; (a_1 \dots a_m)[\nabla/\delta(a_m, c)]\delta(a_m, c).$$

Finally, for the parity condition of this automaton, put

$$\Omega(\alpha\nabla\beta) \;:=\; \begin{cases} 2 \cdot |\beta| + 1 & \text{if } \widetilde{\beta} \notin \mathcal{M}, \\ 2 \cdot |\beta| + 2 & \text{if } \widetilde{\beta} \in \mathcal{M}. \end{cases}$$

In order to prove the equivalence of $\mathbb{A}$ and $\mathbb{A}'$, consider the runs ρ and ρ' of $\mathbb{A}$ and $\mathbb{A}'$, respectively, on some C-stream σ. Let $Q = \mathit{Inf}(\rho)$ denote the set of states of $\mathbb{A}$ that are visited infinitely often during ρ. Clearly then from a certain moment on, ρ will *only* pass states in Q. It is not so hard to see that from that same moment, ρ' will only pass states of the form $\alpha \triangledown \beta$ with $\widetilde{\beta} \subseteq Q$. In other words, we have

$$\mathit{Inf}(\rho') \subseteq \{\alpha \triangledown \beta \mid |\beta| \leq |Q|\}.$$

But also, since Q consists exactly of the states that ρ passes infinitely often, ρ' will infinitely often pass states $\alpha \triangledown \beta$ with $\widetilde{\beta} = Q$. Clearly, among the ones in the right hand side of (12), these will be the ones with the *longest* β-part, and hence, the *highest* parity. From this it follows that

$$\text{the highest parity of the states in } \mathit{Inf}(\rho') \text{ is } \textit{even} \text{ iff } Q \in \mathcal{M}.$$

But this then clearly shows that ρ' is accepting iff ρ is accepting, which suffices to prove the equivalence of $\mathbb{A}$ and $\mathbb{A}'$. QED

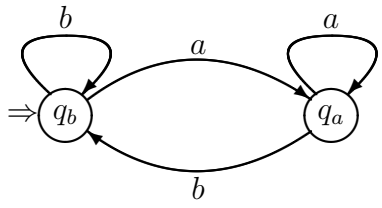
The following example shows that, in the case of deterministic stream automata, the recognizing power of Muller and parity automata is *strictly* stronger than that of Büchi automata.

Example 5.13 Consider the following language over the alphabet $C = \{a, b\}$:

$$L = \{\alpha \in C^\omega \mid a \notin \mathit{Inf}(\alpha)\}.$$

That is, L consists of those C-streams in which the symbol a occurs at most finitely often.

It is not difficult to see that there is a deterministic Muller automaton recognizing this language. Consider the automaton $\mathbb{A}_2$ given by the following diagram,



and Muller acceptance condition $\mathcal{M}_2 := \{\{q_b\}\}$. It is straightforward to verify that the run of $\mathbb{A}_2$ on an $\{a, b\}$-stream α keeps circling in q_b iff from a certain moment on, α only produces b's.

However, there is *no* deterministic *Büchi* automaton recognizing L. Suppose for contradiction that $L = L_\omega(\mathbb{A})$, where $\mathbb{A} = \langle A, \delta, F, a_I \rangle$ is some Büchi automaton. Since the stream $\alpha_0 = bbb\ldots$ belongs to L, it is accepted by $\mathbb{A}$. Hence in particular, the run ρ_0 of $\mathbb{A}$ on α_0 will pass some state $f_0 \in F$ after a finite number, say n_0, of steps.

Now consider the stream $\alpha_1 = b^{n_0}abbb\ldots$. Since runs are uniquely determined, the initial n_0 steps of the run ρ_1 of $\mathbb{A}$ on α_1 are identical to the first n_0 steps of $\mathbb{A}$ on α_0. But since α_1 belongs to L too, it too is accepted by $\mathbb{A}$. Thus on input α_1, $\mathbb{A}$ will visit a state in F infinitely often. That is, we may certainly choose an $n_1 \geq 1$ such that ρ_1 passes some state $f_1 \in F$ after $n_0 + n_1$ steps. Now consider the stream $\alpha_2 = b^{n_0}ab^{n_1}abbb\ldots$, and analyze the run ρ_2 of $\mathbb{A}$ on α_2. Continuing like this, we can find positive numbers $n_0, n_1, \ldots$ such that for every $k \in \omega$, the stream

$$\alpha_k = b^{n_0}ab^{n_1}\ldots ab^{n_k}abbb\ldots \in L, \text{ for all } k. \tag{12}$$

Consider the stream

$$\alpha = (b^{n_0}a)(b^{n_1}a)\ldots(b^{n_k}a)\ldots$$

Containing infinitely many a's, α does not belong to L. Nevertheless, it follows from (12) that the run ρ of $\mathbb{A}$ on α passes through the states $f_0, f_1, \ldots$ as described above. Since F is finite, there is then at least one $f \in F$ appearing infinitely often in this sequence. Thus we have found an $f \in F$ that is passed infinitely often by ρ, showing that $\mathbb{A}$ accepts α. This gives the desired contradiction. ◁

5.2 Nondeterministic automata

Nondeterministic automata generalize deterministic ones in that, given a state and a color, the next state is not *uniquely* determined, and in fact need not exist at all.

Definition 5.14 Given an *alphabet* C, a *nondeterministic* C*-automaton* is a quadruple $\mathbb{A} = \langle A, \Delta, Acc, a_I\rangle$, where A is a finite set, $a_I \in A$ is the *initial state* of $\mathbb{A}$, $\Delta : A \times C \to \wp(A)$ its *transition function* of $\mathbb{A}$, and $Acc \subseteq A$ its *acceptance condition*. ◁

As a consequence, the run of an nondeterministic automaton on a stream is no longer uniquely determined either.

Definition 5.15 A *run* of a deterministic automaton $\mathbb{A} = \langle A, \Delta, Acc, a_I\rangle$ on an C-stream $\gamma = c_0c_1c_2$ is an infinite A-sequence

$$\rho = a_0a_1a_2\ldots$$

such that $a_0 = a_I$ and $a_{i+1} \in \Delta(a_i, c_i)$ for every $i \in \omega$. ◁

Now that runs are no longer unique, an automaton may have both successful and unsuccessful runs on a given stream. Consequently, there is a choice to made concerning the notion of acceptance.

Definition 5.16 A nondeterministic C-automaton $\mathbb{A} = \langle A, \Delta, Acc, a_I\rangle$ *accepts* an C-stream γ if there is a successful run of $\mathbb{A}$ on γ. ◁

Further concepts, such as the language recognized by an automaton, the notion of equivalence of two automata, and the Büchi, Muller and parity acceptance conditions, are defined as for deterministic automata. Also, the transformations given in the Propositions 5.10, 5.11 and 5.12 are equivalence-preserving for nondeterministic automata just as for deterministic one. *Different* from the deterministic case, however, is that *nondeterministic* Büchi automata have the *same* accepting power as their Muller and parity variants.

Proposition 5.17 *There is an effective procedure transforming a nondeterministic Muller stream automaton into an equivalent nondeterministic Büchi stream automaton.*

Proof. Let $\mathbb{A} = \langle A, \Delta, \mathcal{M}, a_I \rangle$ be a nondeterministic Muller automaton. The idea underlying the definition of the Büchi equivalent $\mathbb{A}'$ is that $\mathbb{A}'$, while copying the behavior of $\mathbb{A}$, *guesses* the set $Inf(\rho)$ of a successful run of $\mathbb{A}$, and at a certain (nondeterministically chosen) moment confirms this choice by moving to a position of the form $(a, M, \varnothing)$. In order to make sure that not too many streams are accepted, the device has to keep track which of the states in M have been visited by $\mathbb{A}$, resetting this counter to the empty set every time when *all* M-states have been passed.

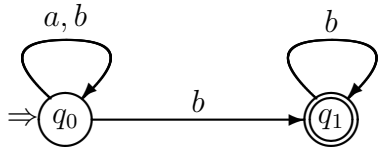
$$\begin{array}{rcl}
A' & := & A \cup \bigcup_{M \in \mathcal{M}} \{(a, M, B) \mid a \in M, B \subseteq M\}, \\
a'_I & := & a_I \\
\Delta(a, c) & := & \Delta(a, c) \cup \bigcup_{M \in \mathcal{M}} \{(b, M, \varnothing) \mid b \in \Delta(a, c) \cap M\} \\
\Delta((a, M, B), c) & := & \begin{cases} \{(b, M, B \cup \{a\}) \mid b \in \Delta(a, c) \cap M\} & \text{if } B \neq M, \\ \{(b, M, \{a\}) \mid b \in \Delta(a, c) \cap M\} & \text{if } B = M, \end{cases} \\
F & := & \{(a, M, B) \in A' \mid B = M\}.
\end{array}$$

We leave it as an exercise for the reader to verify that the thus constructed automaton is indeed equivalent to $\mathbb{A}$. QED

Example 5.18 For a nondeterministic Büchi automaton recognizing the language

$$L = \{\alpha \in C^\omega \mid a \notin Inf(\alpha)\}$$

of Example 5.13, consider the automaton given by the following picture:



In general, the Büchi acceptance condition $F \subseteq A$ of an automaton $\mathbb{A}$ is depicted by the set of states with *double circles*. So in this case, $F = \{q_1\}$. $\lhd$

As a consequence of Proposition 5.17, and witnessed by the Examples 5.13 and 5.18, the recognizing power of nondeterministic Büchi automata is strictly greater than that of their deterministic variants. A key result in automata theory states that when we turn to Muller and parity automata, nondeterminism does *not* increase recognizing power.

Theorem 5.19 *There is an effective procedure transforming a nondeterministic Büchi stream automaton into an equivalent deterministic Muller stream automaton.*

Proof.

▶ `Proof using Safra construction to be supplied`

QED

We may summarize the relative power of the automata concept in the diagram below. Arrows indicate the reducibility of one concept to another, 'D' and 'ND' are short for 'deterministic' and 'nondeterministic', respectively.

$$\begin{array}{ccccc} \text{D Büchi} & \Longrightarrow & \text{D Muller} & \Longrightarrow & \text{D parity} \\ \Downarrow & & \Updownarrow & & \Updownarrow \\ \text{ND Büchi} & \Longleftrightarrow & \text{D Muller} & \Longleftrightarrow & \text{ND parity} \end{array}$$

5.3 A coalgebraic perspective

In this section we introduce a coalgebraic perspective on streams and stream automata. We have two reasons for doing so. First, we hope that this coalgebraic presentation will facilitate the introduction of automata operating on different kinds of structures, in particular, the Kripke automata of the next Chapter. And second, we also believe that the coalgebraic perspective, in which the similarities between automata and the objects they classify comes out more clearly, makes it easier to understand some of the fundamental concepts and results in the area.

Definition 5.20 A *C-stream coalgebra* is a pair $\mathbb{S} = \langle S, \sigma \rangle$ with $\sigma : S \to C \times S$. If we add an (initial) state $s_0 \in S$ to such a structure, we obtain a *pointed C-stream coalgebra*. $\lhd$

Clearly then, streams over an alphabet C can be seen as pointed C-stream coalgebras: simply identify the word $\gamma = c_0c_1c_2\ldots$ with the structure $\langle \omega, 0, \lambda n.(c_n, n+1)\rangle$. Conversely, with any pointed stream coalgebra $\mathbb{S} = \langle S, s_0, \sigma\rangle$ we may associate a unique word $\gamma_{\mathbb{S}}$ by inductively defining $s_{i+1} := \sigma_1(s_i)$, and putting $\gamma_{\mathbb{S}}(n) := \sigma_0(s_n)$. In the sequel, we will swap between these two kinds of objects without explicit notice.

It will be instructive to define the following notion of equivalence between stream coalgebras. As its name already indicates, we are dealing with the analog of the notion of a bisimulation between two Kripke models. Since stream coalgebras, having a deterministic transition structure, are less complex objects than Kripke models, the notion of bisimulation is also, and correspondingly, simpler.

Definition 5.21 Let $\mathbb{S}$ and $\mathbb{S}'$ be two C-stream coalgebras. Then a nonempty relation $Z \subseteq S \times S'$ is a *bisimulation* if the following holds, for every $(s, s') \in Z$:

(color) $\sigma_0(s) = \sigma'_0(s')$;

(successor) $(\sigma_1(s), \sigma'_1(s')) \in Z$.

Two pointed coalgebras $(\mathbb{S}, s)$ and $(\mathbb{S}', s')$ are called *bisimilar*, notation: $\mathbb{S}, s \leftrightarrows \mathbb{S}', s'$ if there is some bisimulation Z linking s to s'. In case the coalgebras $\mathbb{S}$ and $\mathbb{S}'$ are implicitly understood, we may drop reference to them and simply call s and s' bisimilar. $\lhd$

It is not difficult to verify that among 'real' C-streams, (i.e., stream coalgebras stemming from maps $\omega \to C$), bisimilarity means *identity*.

Definition 5.22 A stream is called *regular* if it is bisimilar to a finite stream coalgebra. $\lhd$

Associated is a new perspective on nondeterministic stream automata which makes them very much *resemble* these stream coalgebras. Roughly speaking the idea is this. Think of establishing a bisimulation between two coalgebras in terms of one pointed coalgebra $\mathbb{A} = \langle A, a_I, \alpha\rangle$ *classifying* the other, $\mathbb{S} = \langle S, s_0, \sigma\rangle$.

Now on the one hand make a restriction in the sense that the classifying coalgebra $\mathbb{A}$ must be finite, but on the other hand, instead of demanding its transition function to be of the form $\alpha : A \to C \times A$, allow objects $\alpha(a)$ to be *sets* of pairs in $C \times A$, rather than single pairs. That is, introduce *non-determinism* by letting the transition map Δ of $\mathbb{A}$ be of the form

$$\Delta : A \to \wp(C \times A).$$

Remark 5.23 This presentation is completely *equivalent* to the one given earlier. The point is that there is a natural bijection between maps of the above kind, and the ones given in Definition 5.14 as the transition structure of nondeterministic automata:

$$A \to \wp(C \times A) \;\cong\; (A \times C) \to \wp(A). \tag{13}$$

To see why this is so, an easy proof suffices. Using the principle of currying (as in the proof of Proposition 2.12), we can show that

$$A \to ((C \times A) \to 2) \cong (A \times C \times A) \to 2 \cong (A \times C) \to (A \to 2),$$

where the first and last set can be identified with respectively the left and right hand side of (13) using the bijection between subsets and their characteristic functions. $\lhd$

Thus we arrive at the following reformulation of the definition of nondeterministic automata. Note that with this definition, a stream automaton can be seen as a kind of 'multi-stream' in the sense that every state harbours a *set* of potential 'local realizations' as a stream coalgebra. Apart from this, an obvious difference with stream coalgebras is that stream automata also have an acceptance condition.

Definition 5.24 A *nondeterministic C-stream automaton* is a quadruple $\mathbb{A} = \langle A, \Delta, Acc, a_I \rangle$ such that $\Delta : A \to \wp(C \times A)$ is the *transition function*, $Acc \subseteq A^\omega$ is the *acceptance condition*, and $a_I \in A$ is the *initial state* of the automaton. $\lhd$

Finally, it makes sense to formulate the notion of an automaton *accepting* a stream coalgebra in terms that are related to that of establishing the existence of a bisimulation. The nondeterminism can nicely be captured in game-theoretic terms — note however, that here we are dealing with a single player only.

In fact, bisimilarity between two pointed stream coalgebras can itself be captured game-theoretically, using a trivialized version of the bisimilarity game for Kripke models of Definition 2.16. Consider two stream coalgebras $\mathbb{A}$ and $\mathbb{S}$. Then the *bisimulation game* $\mathcal{B}(\mathbb{A}, \mathbb{S})$ between $\mathbb{A}$ and $\mathbb{S}$ is defined as a board game with positions of the form $(a, s) \in A \times S$, all belonging to $\exists$. At position (a, s), if a and s have a different color, $\exists$ looses immediately; if on the other hand $\alpha_0(a) = \sigma_0(s)$, then as the next position of the match she 'chooses' the pair consisting of the successors of a and s, respectively. These rules can concisely be formulated as in the following Table:

Position	Player	Admissible moves
$(a, s) \in A \times S$	$\exists$	$\{(\alpha_1(a), \sigma_1(s)) \mid \alpha_0(a) = \sigma_0(s)\}$

Finally, the winning conditions of the game specify that $\exists$ wins all infinite games. We leave it for the reader to verify that a pair $(a, s) \in A \times S$ is a winning position for $\exists$ iff a and s are bisimilar.

In order to proceed, however, we need to make a slight modification. We add positions of the form $(\alpha, s) \in (C \times A) \times S$, and insert an 'automatic' move immediately after a basic position, resulting in the following Table.

Position	Player	Admissible moves
$(a, s) \in A \times S$	-	$\{(\alpha(a), s)\}$
$(\alpha, s) \in (C \times A) \times S$	$\exists$	$\{(\alpha_1, \sigma_1(s)) \mid \alpha_0 = \sigma_0(s)\}$

The acceptance game of a nondeterministic automaton $\mathbb{A}$ and a stream coalgebra $\mathbb{S}$ can now be formulated as a natural generalization of this game.

Definition 5.25 Given a nondeterministic C-stream automaton $\mathbb{A} = \langle A, a_I, \Delta, Acc\rangle$ and a pointed stream coalgebra $\mathbb{S} = \langle S, s_0, \sigma\rangle$, we now define the *acceptance game* $\mathcal{A}(\mathbb{A}, \mathbb{S})$ as the following board game.

Position	Player	Admissible moves
$(a, s) \in A \times S$	$\exists$	$\{(\alpha, s) \in (C \times A) \times S \mid \alpha \in \Delta(a)\}$
$(\alpha, s) \in (C \times A) \times S$	$\exists$	$\{(\alpha_1, \sigma_1(s)) \mid \alpha_0 = \sigma_0(s)\}$

Table 5: Acceptance game for nondeterministic stream automata

Its positions and rules are given in Table 5, whereas the winning conditions of infinite matches are specified as follows. Given an infinite match of this game, first select the sequence

$$(a_0, s_0)(a_1, s_1)(a_2, s_2)\dots$$

of *basic positions*, that is, the positions reached during play that are of the form $(a, s) \in A \times S$. Then the match is winning for $\exists$ if the 'A-projection' $a_0a_1a_2\dots$ of this sequence belongs to Acc. $\lhd$

Definition 5.26 A nondeterministic C-stream automaton $\mathbb{A} = \langle A, a_I, \Delta, Acc\rangle$ *accepts* a pointed stream coalgebra $\mathbb{S} = \langle S, s_0, \sigma\rangle$ if the pair (a_I, s_0) is a winning position for $\exists$ in the game $\mathcal{A}(\mathbb{A}, \mathbb{S})$. $\lhd$

5.4 Alternation

In the previous section we saw that we can model the acceptance procedure of a nondeterministic automaton as a single player game. This immediately begs the question why not to allow some interaction in the form of a second player. We thus arrive at a fundamental concept from theoretical computer science, viz., that of machine models based on *alternation*. Very roughly, the idea underlying the alternating machine model is that apart from *existential* choices made by the player that is working towards a successful run of the machine, there are also *universal* choices yielding parallel runs *each* of which have to be successful by the machine. For a more precise formulation of the concept, a game-theoretic framework is the best context. For instance, game theory allows us to naturally generalize the notion of a *run* of a machine on an input object, to that of a *match* being played in order to determine the behavior of a machine on a given input object.

Before we can move to the definition of alternating stream automata, we first need to look at the details of how to represent players' choices in acceptance games. In these notes we will consider the following two approaches:

set-theoretic : represent the choice of a player as a set of options, as in Definition 5.24;

logical : represent choices using connectives (disjunctions for $\exists$, conjunctions for $\forall$).

The two approaches are in fact interchangeable. In one direction, this is easy to see. For instance, we might have chosen to define nondeterministic automata as structures with a transition function mapping a state a of the automaton to a *disjunction* of pairs in $C \times B$. In this set-up we would represent a set $\Delta(a) = \{(\alpha_1), \ldots, \alpha_k\}$ by one of the terms $(\alpha_1 \vee \cdots \vee \alpha_k)$ or $\bigvee_{1 \leq i \leq k} \alpha_i$. In case $\Delta(a) = \varnothing$ this would yield the term $\bot = \bigvee \varnothing$.

The disadvantage of the set-theoretic approach is that one has to specify explicitly to which player the set of options belongs. Hence for the time being we will focus on the logical approach.

Definition 5.27 Given set X, let $\mathit{SLatt}(X)$ denote the set of finite disjunctions (semilattice terms) of elements of X:

$$\varphi ::= x \in X \mid \bigvee \Phi$$

whereas $\mathit{Latt}(X)$ denotes the set of all finite lattice terms of elements of X:

$$\varphi ::= x \in X \mid \bigvee \Phi \mid \bigwedge \Phi$$

Here Φ denotes a finite set of semilattice terms (lattice terms, respectively). $\lhd$

Definition 5.28 An *logical C-stream automaton* is a quadruple $\mathbb{A} = \langle A, \Delta, \mathit{Acc}, a_I \rangle$ such that $\Delta : A \to \mathit{Latt}(C \times A)$ is the *transition function*, $\mathit{Acc} \subseteq A^\omega$ is the *acceptance condition*, and $a_I \in A$ is the *initial state* of the automaton. $\lhd$

Given this definition it is fairly obvious how to define acceptance.

Definition 5.29 Given a logical C-stream automaton $\mathbb{A} = \langle A, a_I, \Delta, \mathit{Acc} \rangle$ and a stream coalgebra $\mathbb{S} = \langle S, \sigma \rangle$, the *acceptance game* $\mathcal{A}(\mathbb{A}, \mathbb{S})$ is given by the rules of Table 6, together with the winning conditions that specify that the winner of an infinite match is determined by the projection on $\mathbb{A}$ of the sequence of basic positions $(a, s) \in A \times S$ in the match, using Acc. $\lhd$

We needed to add automatic moves at basic positions. Since it is not a priori known whether $\Delta(a)$ is a conjunction or a disjunction, we cannot assign a position of the form (a, s) to one of the players.

Definition 5.30 A logical C-stream automaton $\mathbb{A} = \langle A, a_I, \Delta, \mathit{Acc} \rangle$ *accepts* a pointed stream coalgebra $\mathbb{S} = \langle S, s_0, \sigma \rangle$, if the pair (a_I, s_0) is a winning position for $\exists$ in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$. $\lhd$

Position	Player	Admissible moves
$(a, s) \in A \times S$	$-$	$\{(\Delta(a), s)\}$
$(\bigvee \Phi, s) \in \mathit{Latt}(C \times A)$	$\exists$	$\{(\varphi, s) \mid \varphi \in \Phi\}$
$(\bigwedge \Phi, s) \in \mathit{Latt}(C \times A)$	$\forall$	$\{(\varphi, s) \mid \varphi \in \Phi\}$
$(\alpha, s) \in (C \times A) \times S$	$\exists$	$\{(\alpha_1, \sigma_1(s)) \mid \alpha_0 = \sigma_0(s)\}$

Table 6: Acceptance game for logical stream automaton

▶ `Examples to be added`

We will now discuss how to give a set-theoretic definition of alternating automaton. These will be based on a *distributive normal form* that we may develop for logical alternating automata.

Definition 5.31 Two lattice terms φ and ψ are *equivalent* if the equation $\varphi \approx \psi$ holds in the variety of distributive lattices. $\lhd$

Alternatively, φ and ψ are equivalent if they are equivalent as (classical) propositional formulas.

Proposition 5.32 *Let $\mathbb{A} = \langle A, a_I, \Delta, \mathit{Acc}\rangle$ and $\mathbb{A}' = \langle A, a_I, \Delta', \mathit{Acc}\rangle$ be two logical stream automata such that for all $a \in A$, $\Delta(a)$ and $\Delta'(a)$ are equivalent. Then $\mathbb{A}$ and $\mathbb{A}'$ are equivalent automata.*

▶ `Proof to be added`

As a corollary, every logical alternating automaton $\mathbb{A} = \langle A, a_I, \Delta, \mathit{Acc}\rangle$ can be brought into distributive normal form, namely, by rewriting every $\Delta(a)$ as an equivalent disjunction of conjunctions of atomic terms. But such a disjunction of conjunctions can also be represented as a set of sets:

$$\bigvee_{i \in I} \bigwedge_{j \in J_i} \varphi_{i_j} \;\cong\; \Big\{\{\varphi_{i_j} \mid j \in J_i\} \mid i \in I\Big\}$$

This suggests to represent the transition function of an alternating stream automaton as a map

$$\Delta : A \to \wp\wp(C \times A).$$

Here the first power set symbol represents a choice for $\exists$, and the second one, a choice for $\forall$.

Convention 5.33 In the sequel, when giving the transition function of an automaton in set-theoretical format, we will frequently mention the player explicitly whose choice is represented. In particular, we will write $\wp_P(S)$ to indicate the collection of subsets of S, with the understanding that at any position involving such a set $X \subseteq S$, this position belongs to P, that P's move consists of choosing an element x of X, and that the next position is obtained by replacing X with this chosen element x.

Thus we will often write, for instance,

$$\Delta : A \to \wp_\exists \wp_\forall (C \times A).$$

for the transition map of an alternating automaton.

This brings us to the following definition.

Definition 5.34 An *alternating C-stream automaton* is a quadruple $\mathbb{A} = \langle A, \alpha, Acc, a_I \rangle$ such that $\Delta : A \to \wp_\exists \wp_\forall (C \times A)$ is the *transition function*, $Acc \subseteq A^\omega$ is the *acceptance condition*, and $a_I \in A$ is the *initial state* of the automaton.

The admissible moves of the *acceptance game* associated with these automata are given in Table 7, and its winning conditions are standardly derived from the acceptance condition Acc. An alternating C-stream automaton $\mathbb{A} = \langle A, a_I, \Delta, Acc \rangle$ *accepts* a pointed stream coalgebra $\mathbb{S} = \langle S, s_0, \sigma \rangle$, if the pair (a_I, s_0) is a winning position for $\exists$ in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$. ◁

Position	Player	Admissible moves
$(a, s) \in A \times S$	$\exists$	$\{(\Gamma, s) \in \wp_\forall (C \times A) \times S \mid \Gamma \in \Delta(a)\}$
$(\Gamma, s) \in \wp_\forall (C \times A) \times S$	$\forall$	$\{(\gamma, s) \in (C \times A) \times S \mid \gamma \in \Gamma\}$
$(\gamma, s) \in (C \times A) \times S$	$\exists$	$\{(\gamma_1, \sigma_1(s)) \mid \gamma_0 = \sigma_0(s)\}$

Table 7: Acceptance game for alternating stream automata

6 Graph automata

In this chapter we introduce and discuss the automata that are used to study the modal μ-calculus. These *graph* automata all operate on the same type of structures, namely pointed Kripke models. Nevertheless, they come in a large variety of shapes, so it is good to observe that roughly speaking, every graph automaton belongs to either of the following two kinds.

1. First we introduce *modal automata*; these represent fairly straightforward generalizations of modal fixpoint formulas in μPML.

2. We then consider *Kripke automata*, which closely resemble the pointed Kripke structures on which they are supposed to operate.

Of these two kinds, the reader may at first sight find the first one more intuitive and easy to understand — this is also the reason why they are introduced first.

However, the Kripke automata are far more important since they allow us to prove strong results about the modal μ-calculus. In particular, a key result, here given as Theorem 6.17, states that every alternating Kripke automaton can be effectively transformed into a nondeterministic equivalent. As we will see in Chapter 9, many crucial results concerning the modal μ-calculus, such as its decidability and small model property, are direct consequences of this fundamental theorem.

Finally, it should be stressed that the second kind of automaton *also* generalizes fixpoint formulas, namely, the ones that use the coalgebraic connective $\bullet$ of the sections 2.6 and 3.5 rather than the standard boxes and diamonds.

Convention 6.1 Throughout this chapter we define automata that are supposed to accept or reject pointed Kripke models. In each case, this automaton is of the form $\mathbb{A} = \langle A, a_I, \Delta, \Omega \rangle$ where A is a set of states, $a_I \in A$ is the initial state, Δ is some kind of transition function on A, and $\Omega : A \to \omega$ is a parity condition.

Also in each case, the question whether such an automaton accepts or rejects a given pointed Kripke model $(\mathbb{S}, s)$ is determined by playing some kind of *acceptance game*. This game will always proceeds in *rounds*, from one basic position $(a, s) \in A \times S$ via some intermediate position(s) to a new basic position. The rules of this game are determined by the precise shape of the transition function Δ, and in each case will be given explicitly. However, the winning conditions are fixed. Finite matches, as always, are lost by the player who got stuck. The winner of an infinite match β is always determined by the infinite sequence $(a_I, s)(a_1, s_1)(a_2, s_2) \ldots$ of basic positions occurring in β, using the parity condition Ω on the sequence $a_I a_1 a_2 \ldots$

Finally, the definition of acceptance is also fixed: the automaton $\mathbb{A}$ *accepts* the pointed Kripke model $(\mathbb{S}, s)$ precisely if the pair (a_I, s) is a winning position for $\exists$ in the acceptance game.

Definition 6.2 Let $\mathbb{A}$ be some kind of automaton for pointed Kripke models. The class of pointed Kripke models that are accepted by a given automaton $\mathbb{A}$ is denoted as $\mathsf{L}(\mathbb{A})$. $\lhd$

Unless explicitly specified otherwise, throughout this chapter we work with a fixed set P of propositional variables, and a fixed set D of atomic actions. For notational convenience, we will usually abbreviate the associated Kripke functor (cf. Definition 2.10) $K_{\mathsf{D},\mathsf{P}}$ by K.

6.1 Modal automata

Formulas and automata are very much alike. As any reader going through the chapters 3 and 5 will have observed, there are many resemblances between the evaluation game of a modal (fixpoint) formula and the acceptance game of an automaton. States of the automata seem to have their counterpart in the *bound* variables of the formula, with greatest and least fixpoint operators corresponding to even and odd parity, respectively.

Of course, an important difference is that in the case of a modal formula, interaction between players is not restricted to the Boolean connectives. Unlike in streams, successor states in Kripke models are not unique, and the game reflects this by allowing, depending on the modal connective, one of the players to chooses a next state in the Kripke model. But there is in fact no reason why we could not incorporate this kind of interaction in the definition of automata for Kripke models.

Definition 6.3 Given a set A and a Kripke functor $K = K_{\mathsf{D},\mathsf{P}}$, the set $\mathit{MLatt}_K(A)$ of *K-modal lattice terms over A* is inductively defined as follows:

$$\varphi ::= p \mid \neg p \mid \langle d \rangle a \mid [d]a \mid a \mid \bot \mid \top \mid \varphi \vee \varphi \mid \varphi \wedge \varphi.$$

Here p, a and d refer to arbitrary elements of P, A, and D, respectively. $\lhd$

Definition 6.4 A *modal automaton over* P is an automaton $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ such that $\Delta : A \to \mathit{MLatt}_K(A)$.

The acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ associated with such an automaton $\mathbb{A}$ and a pointed Kripke model $(\mathbb{S}, s)$ is determined by the rules given in Table 8 (given Convention 6.1). $\lhd$

As will become clear from Theorem 6.6 and its proof, perhaps the best perspective on these modal automata is that they *generalize* the modal fixpoint formulas of Chapter 3. The basic idea is to be more liberal concerning structure: while formulas by definition are required to have a tree structure (with back edges representing the unfolding relation between a fixpoint variable and its unfolding formula), for automata we accept structures that allow cyclicity.

▶ `examples to be added`

Position	Player	Admissible next moves
$(a, s) \in A \times S$	–	$\{(\Delta(a), s)\}$
$(\varphi_1 \vee \varphi_2, s) \in \mathit{MLatt}_K(A)$	$\exists$	$\{(\varphi_1, s), (\varphi_2, s)\}$
$(\varphi_1 \wedge \varphi_2, s) \in \mathit{MLatt}_K(A)$	$\forall$	$\{(\varphi_1, s), (\varphi_2, s)\}$
$(\bot, s) \in \mathit{MLatt}_K(A)$	$\exists$	$\varnothing$
$(\top, s) \in \mathit{MLatt}_K(A)$	$\forall$	$\varnothing$
(p, s), with $p \in \mathsf{P}$ and $s \in V(p)$	$\forall$	$\varnothing$
(p, s), with $p \in \mathsf{P}$ and $s \notin V(p)$	$\exists$	$\varnothing$
$(\neg p, s)$, with $p \in \mathsf{P}$ and $s \in V(p)$	$\exists$	$\varnothing$
$(\neg p, s)$, with $p \in \mathsf{P}$ and $s \notin V(p)$	$\forall$	$\varnothing$
$(\langle d \rangle a, s) \in \mathit{MLatt}_K(A)$	$\exists$	$\{(a, t) \mid t \in \sigma_d(s)\}$
$([d]a, s) \in \mathit{MLatt}_K(A)$	$\forall$	$\{(a, t) \mid t \in \sigma_d(s)\}$

Table 8: Acceptance game for modal automaton

6.2 Formulas and modal automata

Definition 6.5 Let ξ be a formula of the modal μ-calculus, and $\mathbb{A}$ an automaton operating on Kripke models. Then ξ and $\mathbb{A}$ are *equivalent* if

$$\mathbb{S}, s \Vdash \xi \text{ iff } \mathbb{A} \text{ accepts } (\mathbb{S}, s)$$

for every pointed Kripke model $(\mathbb{S}, s)$. $\lhd$

Theorem 6.6 *There is an effective procedure that, given a modal fixpoint formula ξ, returns a modal automaton $\mathbb{A}_\xi$ that is equivalent to ξ.*

Proof. $\mathbb{A}_\xi$ is directly based on the formula structure of ξ, that we may without loss of generality assume to be clean. As usual we let, for a bound variable x of ξ, $\eta_x x.\delta_x$ denote the unique subformula of ξ where x is bound.

For the states of $\mathbb{A}_\xi$ we could take the subformulas of ξ themselves, but the definition may be easier to understand if we make a formal distinction between formulas and states, by putting

$$A := \{\widehat{\varphi} \mid \varphi \in \mathit{Sfor}(\xi)\}.$$

The initial state a_I of $\mathbb{A}_\xi$ will clearly be the state $\widehat{\xi}$.

In order to define the transition function Δ we make a case distinction as to the

kind of subformula that we are dealing with.

$$\begin{array}{lcll}
\Delta(\widehat{\varphi \vee \psi}) & := & \widehat{\varphi} \vee \widehat{\psi} & \\
\Delta(\widehat{\varphi \wedge \psi}) & := & \widehat{\varphi} \wedge \widehat{\psi} & \\
\Delta(\widehat{\chi}) & := & \chi & \text{for } \chi \in \{\top, \bot, p, \neg p\} \\
\Delta(\widehat{\langle d \rangle \varphi}) & := & \langle d \rangle \widehat{\varphi} & \\
\Delta(\widehat{[d]\varphi}) & := & [d]\widehat{\varphi} & \\
\Delta(\widehat{\eta x.\delta}) & := & \widehat{\delta} & \\
\Delta(\widehat{x}) & := & \widehat{\delta_x} &
\end{array}$$

Readers having worries concerning this definition should realize that correctness is in fact trivial — this is *not* an inductive definition!

The acceptance condition of modal automata is given by a parity function Ω that we define now. The only subformulas of which the parity will be of interest are the variables that may get unfolded in the acceptance game for ξ. That is, unless φ is a bound variable of ξ, we put $\Omega(\widehat{\varphi}) := 0$.

This leaves the task of defining of $\Omega(\widehat{x})$ where $x \in BV(\xi)$. Recall that $\leq_\xi$ is the dependency order on these bound variables. It is in fact easy to define a function Ω such that

- $\Omega(\widehat{x})$ is odd if x is a μ-variable, and even if x is a ν-variable, and
- $\Omega(\widehat{x}) \leq \Omega(\widehat{y})$ iff $x \leq_\xi y$.

The details of this definition are left as an exercise to the reader.

With this definition it is easy to see that for any Kripke model $\mathbb{S}$, the acceptance game $\mathcal{A}$ for $\mathbb{A}_\xi$ and $\mathbb{S}$ on the one hand, and the evaluation game $\mathcal{E}$ for ξ and $\mathbb{S}$ on the other, are very similar. It is in fact not hard to prove that for any state s of $\mathbb{S}$, and for any subformula φ of ξ, $(\varphi, s) \in \mathrm{Win}_\exists(\mathcal{E})$ iff $(\widehat{\varphi}, s) \in \mathrm{Win}_\exists(\mathcal{A})$. QED

Theorem 6.7 *There is an effective procedure that, given a modal automaton $\mathbb{A}$, returns a modal fixpoint formula $\xi_\mathbb{A}$ that is equivalent to $\mathbb{A}$.*

Proof. The proof of this theorem will be by induction on the 'complexity' of the automaton which we may simply define as the *highest parity* $m_\mathbb{A} := \max(\Omega[A])$ of $\mathbb{A}$.

In the base case of the induction, where $m_\mathbb{A} = 0$, *all* states of $\mathbb{A}$ have parity 0.

▶ `This case will be addressed later.`

Now we turn to the inductive case, where $m_\mathbb{A} > 0$. Let $M \subseteq A$ be the set of states that actually have parity $m_\mathbb{A}$, then M is nonempty, say $M = \{a_1, \ldots, a_k\}$. Without loss of generality we may assume that the initial state a_I of $\mathbb{A}$ does not belong to M.

The main idea of the proof consists of removing the elements of M as *states* in $\mathbb{A}$, but at the same time adding them as *variables*. This makes that every term $\Delta(a)$ is now a well-formed $MLatt(\mathsf{P} \cup M, A \setminus M)$-term, so that the structure

$$\mathbb{A}_M := \langle A \setminus M, a_I, \Delta\restriction_{A\setminus M}, \Omega\restriction_{A\setminus M} \rangle$$

is a modal automaton over $\mathsf{P} \cup M$. Furthermore, for each $i \in \{1, \ldots, k\}$, we will consider the automaton $\mathbb{A}_i := (A \setminus M, a_i, \Delta\restriction_{A\setminus M}, \Omega\restriction_{A\setminus M})$ which is the version of $\mathbb{A}_M$ that has a_i as its initial position.

Obviously, the inductive hypothesis applies to each of these automata. Thus we obtain fixed point formulas $\varphi_M, \varphi_1, \ldots, \varphi_k$, all taking free variables from the set $\mathsf{P} \cup M$, and such that for any $\mathsf{P} \cup M$-model $\mathbb{S}$ and any point s in $\mathbb{S}$, we have that $\mathbb{A}_M$ accepts $(\mathbb{S}, s)$ iff $\mathbb{S}, s \Vdash \varphi_M$, and, for each i, $\mathbb{A}_i$ accepts $(\mathbb{S}, s)$ iff $\mathbb{S}, s \Vdash \varphi_i$.

Clearly then, for any $\mathsf{P} \cup M$-model $\mathbb{S}$, the k-tuple $\overline{\varphi}$ determines a monotone map $[\![\overline{\varphi}]\!]_{\mathbb{S}} : (\mathcal{P}(S))^k \to (\mathcal{P}(S))^k$ given by

$$[\![\overline{\varphi}]\!]_{\mathbb{S},V}(T_1, \ldots, T_k) := ([\![\varphi_1]\!]_{\mathbb{S}[\overline{a}\mapsto\overline{T}]}, \ldots, [\![\varphi_1]\!]_{\mathbb{S}[\overline{a}\mapsto\overline{T}]}).$$

Here $\mathbb{S}[\overline{a} \mapsto \overline{T}]$ denotes the variant of $\mathbb{S}$ with $\mathsf{P} \cup M$-valuation $V[\overline{a} \mapsto \overline{T}]$ given by

$$V[\overline{a} \mapsto \overline{T}](x) := \begin{cases} T_i & \text{if } x = a_i \in M, \\ V(x) & \text{if } x \in \mathsf{P}, \end{cases}$$

where V is the valuation of $\mathbb{S}$.

It follows from standard fixed point theory (cf. the discussion in Appendix A, following Proposition A.10, of the Gaussian elimination method), that the least and greatest fixed points of this map are given by μML-formulas. More precisely, there are formulas $\varphi_1^\mu, \ldots, \varphi_k^\mu$ and $\varphi_1^\nu, \ldots, \varphi_k^\nu$, all with free variables in P, such that

$$([\![\varphi_1^\mu]\!]^{\mathbb{S}}, \ldots, [\![\varphi_k^\mu]\!]^{\mathbb{S}}) \text{ is the } \textit{least} \text{ fixed point of } [\![\overline{\varphi}]\!]^{\mathbb{S}}$$

for every P-model $\mathbb{S}$, and likewise for the greatest fixed point.

Now let $\psi_{\mathbb{A}}$ be the formula

$$\xi_{\mathbb{A}} := \varphi_M[\overline{\varphi}^\eta/\overline{a}].$$

That is, we uniformly substitute, in φ_M, each a_i with the formula φ_i^η, where η denotes μ if $m_{\mathbb{A}}$ is odd, and ν if $m_{\mathbb{A}}$ is even. The proof that this formula $\xi_{\mathbb{A}}$ is indeed equivalent to the automaton $\mathbb{A}$ is fairly similar to the proof of the Adequacy Theorem, whence we omit further details. QED

6.3 Kripke automata

We have now arrived at the introduction of the second kind of automata for Kripke models: the ones that are similar to the Kripke models rather than to standard modal

fixpoint formulas. These automata can be introduced in the same way as the alternating stream automata of subsection 5.4, with the understanding that the notion of bisimulation between Kripke models is far more complicated than that between stream coalgebras.

So consider, for two Kripke models $\mathbb{A} = \langle A, \alpha \rangle$ and $\mathbb{S} := \langle S, \sigma \rangle$, the bisimulation game $\mathcal{B}(\mathbb{A}, \mathbb{S})$ of Definition 2.16. The main conceptual step is to think of $\mathbb{A}$ as a 'proto-automaton' that we use to *classify* $\mathbb{S}$ rather than as of a Kripke model that we are comparing with $\mathbb{S}$. In order to turn $\mathbb{A}$ into a proper Kripke automaton, four technical modifications have to be made:

1. A small change is that we require $\mathbb{A}$ (i.e., its base set A) to be finite — but in fact, it would be perfectly acceptable to allow for infinite automata as well.

2. Second, and equally undramatic, we have to add an initial state to the structure of $\mathbb{A}$.

3. Third, whereas the winner of an infinite match of a bisimulation game is always $\exists$, the winner of an infinite acceptance match will be determined by an explicit acceptance condition on A^ω — a parity condition, in our case.

4. The fourth and foremost modification is that we introduce *nondeterminism*, or even *alternation* to the transition structure of $\mathbb{A}$. Just as for stream automata, Kripke automata will harbour many 'realizations' of Kripke models — and as for stream automata our approach is set-theoretic rather than logical.

Our presentation of alternating Kripke automata is of a set-theoretic nature, and uses the $\wp_\exists/\wp_\forall$ notation of Convention 5.33.

Definition 6.8 Given a Kripke functor K, an *(alternating) Kripke automaton* is a quadruple $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ such that the transition function Δ is given as a map $\Delta : A \to \wp_\exists\wp_\forall(KA)$. Such a Kripke automaton is called *nondeterministic* if, for every $a \in A$, $|\Gamma| \leq 1$ for all $\Gamma \in \Delta(a)$ (that is, $\Delta(a)$ only contains singletons and possibly the empty set).

The *acceptance game* $\mathcal{A}(\mathbb{A}, \mathbb{S})$ associated with a Kripke automaton $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ and a Kripke structure $\mathbb{S}$ is given by Table 9. $\lhd$

For an informal description of the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$, the most important observation is that matches of this game proceed in rounds moving from one basic position to another. Think of a basic position (a, s) as a statement, defended by $\exists$ and attacked by $\forall$, that a and s 'fit well together', or that $\mathbb{A}$, 'as seen from a', is an adequate description of $\mathbb{S}$, 'as seen from s'.

Each round consists of exactly four moves, with interaction pattern $\exists\forall\exists\forall$:

Position	Player	Admissible moves
$(a, s) \in A \times S$	$\exists$	$\{(\Gamma, s) \in \wp_\forall(K(A)) \times S \mid \Gamma \in \Delta(a)\}$
$(\Gamma, s) \in \wp_\forall(K(A)) \times S$	$\forall$	$\{(\gamma, s) \in K(A) \times S \mid \gamma \in \Gamma\}$
$(\gamma, s) \in KA \times S$	$\exists$	$\{Z \subseteq A \times S \mid (\gamma, \sigma(s)) \in \overline{K}Z\}$
$Z \in \wp(A \times S)$	$\forall$	Z

Table 9: Acceptance game for alternating Kripke automata

- At a basic position (a, s), the 'K-successor' $\sigma(s) \in KS$ of s is fixed, but $\exists$ and $\forall$ first have to play a finite (two-move) subgame in order to determine which element will temporarily act as the analogous 'K-successor' α of a. More precisely, the rules of the game are such that first, $\exists$ chooses a subset $\Gamma \subseteq KA$ from $\Delta(a)$, after which $\forall$ chooses, from the set Γ of options, an actual element $\gamma \in KA$. (Since the order of the players is fixed we do not need to add an additional, automatic move from (a, s) to $(\Delta(a), s)$ as would be the case if we had chosen for a logical, rather than a set-theoretic formalization, cf. the discussion at the beginning of subsection 5.4).

- Halfway the round then, play has arrived at a position of the form $(\gamma, s) \in KA \times S$. The players now proceed as in the bisimilarity game for Kripke models. First, $\exists$ chooses a 'local bisimulation' linking γ and s (or rather: γ and $\sigma(s)$), that is, a relation $Z \subseteq A \times S$ such that $(\gamma, \sigma(s)) \in \overline{K}Z$. Spelled out, this means that $\exists$ can only choose such a relation Z if γ is of the form $(\pi, B) \in \wp(\mathsf{P}) \times \wp(A)^{\mathsf{D}}$ with $\pi = \sigma_V(s)$, and that Z has to satisfy the back and forth conditions for each atomic action d, stating that for all $b \in B$ there is $t \in \sigma_d(s)$ with bZt, and vice versa.

  The second, 'dynamic' half of the round, ends with $\forall$ choosing an element from Z. This element is of the form $(b, t) \in A \times S$ and forms the new basic position.

▶ `associate automaton with Kripke model`

In many cases we will need to work with nondeterministic Kripke automata.

Definition 6.9 Given a Kripke functor K, an alternating Kripke automaton $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ is called *nondeterministic* if, for every $a \in A$, all elements of $\Delta(a)$ are in fact *singletons*. ◁

Remark 6.10 To facilitate the presentation, we will often represent the transition function of such a device $\mathbb{A}$ as a map $\delta : A \to \wp_\exists(KA)$. That is, rather than taking a set $\Delta(a)$ consisting of singletons we deal with the elements of those singletons directly. The structure of the associated acceptance game then looks as in Table 10.

◁

Position	Player	Admissible moves
$(a, s) \in A \times S$	$\exists$	$\{(\gamma, s) \in K(A) \times S \mid \gamma \in \delta(a)\}$
$(\gamma, s) \in KA \times S$	$\exists$	$\{Z \subseteq A \times S \mid (\gamma, \sigma(s)) \in \overline{K}Z\}$
$Z \in \wp(A \times S)$	$\forall$	Z

Table 10: Acceptance game for nondeterministic Kripke automata

6.4 Formulas and Kripke automata

Even though these Kripke automata resemble Kripke models much more than the modal automata of section 6.1, we can prove the same equivalence between formulas and automata.

Theorem 6.11 *There is an effective procedure that, given a modal fixpoint formula ξ returns a Kripke automaton $\mathbb{A}_\xi$ that is equivalent to ξ.*

In order to prove this result we need to introduce some automata of 'intermediate' type.

Definition 6.12 A *logical Kripke automaton* or *LKA* is a quadruple $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ with A, a_I and Ω as usual, while $\Delta : A \to \mathit{Latt}(A \cup KA)$.

Such a device is called *guarded* if $\Delta(a) \in \mathit{Latt}(KA)$ for every $a \in A$, and *semiguarded* if $\Omega(a) > \Omega(b)$ whenever $b \lhd a$. Here we define the relation $\lhd \subseteq A \times A$ by putting $b \lhd a$ if and only if b occurs in $\Delta(a)$ (that is, b is an atomic subterm of $\Delta(a)$). $\lhd$

Proof of Theorem 6.11. The proof of Theorem 6.11 is based on the Propositions 6.13, 6.14 and 6.15 below. The basic idea is to turn a modal fixpoint formula into an equivalent Kripke automaton using the following three equivalence preserving transformations:

$$\text{fixpoint formula} \rightsquigarrow \text{semiguarded LKA} \rightsquigarrow \text{guarded LKA} \rightsquigarrow \text{Kripke automaton.}$$

Of these three constructions, the first and third are fairly direct and do not require any new ideas. The only interesting transformation is the one from a semiguarded automaton to a guarded automaton. QED

Proposition 6.13 *There is an effective procedure that, given a formula $\xi \in \mu\mathrm{ML}$, returns an equivalent semiguarded logical Kripke automaton.*

Proof. The proof of this proposition is based on a variation of the ideas underlying the proof of Proposition 3.16. QED

Proposition 6.14 *There is an effective procedure that, given a semiguarded logical Kripke automaton, returns an equivalent guarded one.*

Proof. Let $\mathbb{A} = \langle A, a_I, \Delta, \Omega\rangle$ be a semiguarded logical Kripke automaton. For the definition of its guarded equivalent $\mathbb{A}'$, we need some preparations.

For each state $a \in A$, we will construct, in finitely many steps, a tree $T(a)$, together with a *labelling* and a (partial) *marking* of the nodes of the tree. To set up the construction, we start with the construction tree of the $\mathit{Latt}(A \cup KA)$-term $\Delta(a)$. The inner nodes of this tree are *labelled* with a connective ($\bigvee$ or $\bigwedge$), and the leaves with an *atomic* term of the form $b \in A$ or $\xi \in KA$. (Nodes labelled with terms $\bigvee$ or $\bigwedge$ will be considered as inner nodes also if they have no children.) Furthermore, the root of this tree is *marked* 'a', while all other nodes of the initial tree are marked with the empty list. Now recursively, replace each leaf labelled $b \in A$ with its construction tree, and add 'b' to the mark of the new inner node of the tree. Repeat the process until no leaves are left that are labelled with elements of A, and (thus) all leaves are labelled with elements of KA.

It is not difficult to see that this process must terminate after finitely many steps. The key observation here is that for any two states $a, b \in A$, we find b as the label of some leaf of the construction tree of $\Delta(a)$ if and only if $b \lhd a$. And since $\mathbb{A}$ is semiguarded we have $\Omega(b) > \Omega(a)$ if $b \lhd a$. From this it follows that there are no infinite sequences $a_0 \rhd a_1 \rhd a_2 \rhd \ldots$, and so the algorithm must terminate.

We define $T(a)$ as the tree that is constructed by the algorithm that we just described. Clearly, $T(a)$ can be seen as the construction tree of some $\mathit{Latt}(KA)$ term $\Delta'(a)$. Now consider a match of the game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ which has arrived at a basic position $(a_0, s) \in A \times S$ — or simply the game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ initialized at (a_0, s). The key observation is that $T(a)$ represents the *static* initial part of this game, that is, the part that is played until the automaton moves to a successor of s. The point is that this part of the game is completely determined by the disjuncts and conjuncts chosen by $\exists$ and $\forall$, respectively. More specifically, a maximal path through $T(a)$ corresponds to a partial match of $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_0, s)$ that is *maximal* in the sense that it either ends in a win for one of the players, or else in a position of the form $(\gamma, s) \in KA \times S$, where γ is the label of the leaf corresponding to the last element of the maximal path through $T(a)$.

Now in principle, as the guarded equivalent of $\mathbb{A}$ we would like to take the structure $\underline{\mathbb{A}} := \langle A, a_I, \underline{\Delta}, \Omega\rangle$. Unfortunately, while it would not be hard to see that for any pointed Kripke model $(\mathbb{S}, s)$, the *boards* of the two games $\mathcal{A}(\mathbb{A}, \mathbb{S})$ and $\mathcal{A}(\underline{\mathbb{A}}, \mathbb{S})$ are virtually identical (isomorphic modulo some automatic moves), they are rather different when we look at *parities*. The problem is that in a term $\underline{\Delta}(a)$ many original states of $\mathbb{A}$ are 'hidden', with the effect that their parities go unnoticed when playing the acceptance game for $\underline{\mathbb{A}}$ rather than for $\mathbb{A}$.

To take care of this, with each leaf l of the tree $T(a_0)$ associate, apart from its label $\gamma_l \in KA$, also a unique sequence $a_n \lhd a_{n-1} \lhd \cdots \lhd a_0$, consisting of the marks encountered on the path leading from the root to the leaf l. Since the original automaton $\mathbb{A}$ is *semiguarded*, we find that $\Omega(a_n) > \Omega(a_{n-1}) > \cdots < \Omega(a_0)$. So $\Pi(a_0, l) := \Omega(a_n)$ is the *highest* parity of these a_i — corresponding to the highest parity encountered in

the static partial match of $\mathcal{A}(\mathbb{A}, \mathbb{S})$ from basic position (a_0, s) to the non-basic position (γ_l, s). The key idea in the definition of $\mathbb{A}'$ is to assign this parity to the *children* of γ_l, that is, to the elements of the set B_l where $\gamma_l = (\pi_l, B_l)$. (Here we assume for the moment that we are dealing with the basic case of one single atomic action. In the general case B_l would not be a subset of A but rather a D-indexed collection of such subsets, and we would have to work with the union of these subsets.) However, since elements of A can occur in many of such sets, we cannot base $\mathbb{A}'$ on the state set A of $\mathbb{A}$; rather, we take a copy (a, n) of a for each n in the range $\Omega[A]$ of Ω. Thus we arrive at the following definition.

Given the semiguarded automaton $\mathbb{A} = \langle A, a_I, \Delta, \Omega \rangle$, we define the automaton $\mathbb{A}' = \langle A', a'_I, \Delta', \Omega' \rangle$ as follows:

$$\begin{array}{rcl} A' & := & A \times \Omega[A], \\ a'_I & := & (a_I, \Omega(a_I)), \\ \Omega'(a, n) & := & n, \end{array}$$

while $\Delta'(a, n)$ is the term $\underline{\Delta}(a)$ where, for each leaf l of $T(a)$, the atom $\gamma_l = (\pi_l, B_l) \in KA$ of $\underline{\Delta}(a)$ is replaced with the object $(\pi_l, \{(b, \Pi(a, l)) \mid b \in B_l\}) \in KA'$.

On the basis of the earlier given motivation for this definition, the reader should be able to prove that $\mathbb{A}$ and $\mathbb{A}'$ are indeed equivalent. Furthermore, it is obvious that $\mathbb{A}'$ is a guarded automaton. This suffices to prove the Proposition. QED

Proposition 6.15 *There is an effective procedure that, given a guarded logical Kripke automaton, returns an equivalent Kripke automaton.*

Proof. This proof is straightforward, given the discussion preceding Definition 5.34. QED

Theorem 6.16 *There is an effective procedure that, given a Kripke automaton $\mathbb{A}$, returns a modal fixpoint formula $\xi_{\mathbb{A}}$ that is equivalent to $\mathbb{A}$.*

Proof. The proof of this theorem is completely analogous to that of Theorem 6.7. QED

6.5 A fundamental theorem

The next theorem is perhaps the most fundamental result concerning the automata-theoretic approach towards the modal μ-calculus.

Theorem 6.17 *There is an effective procedure that transforms a given alternating Kripke automaton into an equivalent nondeterministic one.*

▶ `Proof to be supplied`

6.6 Closure properties

In order to discuss the recognizing power of automata, we first need to introduce the notion of a recognizable language.

Definition 6.18 We will refer to a class of pointed Kripke models as a *K-language*. Such a class C is called *recognizable* if there is a Kripke automaton $\mathbb{A}$ such that $\mathsf{C} = \mathsf{L}(\mathbb{A})$. $\lhd$

It immediately follows from earlier results that recognizable K-languages have (at least two) alternative characterizations.

Proposition 6.19 *The following are equivalent for any K-language C:*

1. *C is recognizable;*
2. *C is nondeterministically recognizable, that is, $\mathsf{C} = \mathsf{L}(\mathbb{A})$ for some nondeterministic automaton $\mathbb{A}$;*
3. *C is definable by some modal fixpoint formula.*

Clearly, the recognizing power of a certain type of automaton corresponds to the notion of recognizability. One way to study this notion is to investigate under which class operations the recognizable languages are closed.

Definition 6.20 Let Op be some operation on K-languages, then we say that a class of languages is closed under Op if we obtain a language from this class whenever we apply Op to a family of languages from the class. $\lhd$

It follows immediately from the equivalence $1 \Leftrightarrow 3$ of Proposition 6.19 that the class of recognizable K-languages is closed under taking union, intersection and complementation. However, it is of interest for future reference to have direct, automata-theoretic proofs of the results for union and intersection. For that purpose, we define the sum and product of two K-automata, and prove that they recognize, respectively, the union and the intersection of the languages associated with the original automata.

Definition 6.21 Let $\mathbb{A}_1 = (A_1, a_I^1, \Delta_1, \Omega_1)$ and $\mathbb{A}_2 = (A_2, a_I^2, \Delta_2, \Omega_2)$ be two Kripke automata. We will define their *sum* $\mathbb{A}_\cup$ and *product* $\mathbb{A}_\cap$.

Both of these automata will have the *disjoint union* $A_{12*} := \{*\} \uplus A_1 \uplus A_2$ as their collection of states. Also, the parity function Ω will be the same for both automata:

$$\Omega(a) := \begin{cases} 0 & \text{if } a = *, \\ \Omega_i(a) & \text{if } a \in A_i. \end{cases}$$

The only difference between the automata lies in the transition functions, which are defined as follows:

$$\Delta_\cup(a) := \begin{cases} \Delta_1(a_I^1) \cup \Delta_2(a_I^2) & \text{if } a = * \\ \Delta_i(a) & \text{if } a \in A_i, \end{cases}$$

$$\Delta_\cap(a) := \begin{cases} \{\Phi_1 \cup \Phi_2 \mid \Phi_i \in \Delta_i(a_I^i)\} & \text{if } a = * \\ \Delta_i(a) & \text{if } a \in A_i. \end{cases}$$

Finally, we put $\mathbb{A}_\cup := (A_{12}, *, \Delta_\cup, \Omega)$ and $\mathbb{A}_\cap := (A_{12}, *, \Delta_\cap, \Omega)$. ◁

Proposition 6.22 *Let $\mathbb{A}_1$ and $\mathbb{A}_2$ be two Kripke automata. Then for any pointed K-coalgebra $(\mathbb{S}, s)$ we have:*

1. *$\mathbb{A}_\cup$ accepts $(\mathbb{S}, s)$ iff $\mathbb{A}_1$ or $\mathbb{A}_2$ accepts $(\mathbb{S}, s)$,*
2. *$\mathbb{A}_\cap$ accepts $(\mathbb{S}, s)$ iff both $\mathbb{A}_1$ and $\mathbb{A}_2$ accept $(\mathbb{S}, s)$.*
3. *$\mathbb{A}_\cup$ is non-deterministic if $\mathbb{A}_1$ and $\mathbb{A}_2$ are so.*

Proof. First suppose that the automaton $\mathbb{A}_\cup$ accepts $(\mathbb{S}, s)$. Hence by definition, $\exists$ has a winning strategy f in the acceptance game $\mathcal{A}(\mathbb{A}_\cup, \mathbb{S})$ starting from position $(*, s)$. Let i be such that $f(*, s) \in \Delta(a_I^i)$. It is then straightforward to verify that f, restricted to $\exists$'s positions in $\mathcal{A}(\mathbb{A}_i, \mathbb{S})$, is a winning strategy for $\exists$ from position (s, a_I^i). From this it is immediate that $\mathbb{A}_i$ accepts $(\mathbb{S}, s)$.

The other statements of the proof admit similarly straightforward proofs. QED

Definition 6.23 Let $\mathsf{P}' \subseteq \mathsf{P}$ be two sets of proposition letters, and let $\mathbb{S} = \langle S, \sigma_V, \sigma_R \rangle$ be a Kripke model over P. Then we let $\mathbb{S}\restriction_{\mathsf{P}'}$ denote the restriction of $\mathbb{S}$ to P', that is, the structure $\langle S, \sigma'_V, \sigma_R \rangle$ with $\sigma'_V : S \to \wp(\mathsf{P}')$ given by $\sigma'_V(s) := \sigma_V(s) \cap \mathsf{P}'$. ◁

In the remainder of the section we discuss closure of recognizable Kripke languages under projection (modulo bisimulation). We need to work with nondeterministic automata here. As discussed in Remark 6.10, we will represent the transition function of such a device $\mathbb{A}$ as a map $\delta : A \to \wp_\exists(KA)$.

Definition 6.24 Let P' and P be two sets of proposition letters such that P' is a proper subset of P, and let $\mathbb{A} = \langle A, a_I, \Delta, \Omega \rangle$ be a $K_{\mathsf{P},\mathsf{D}}$-automaton.

Then we define the $K_{\mathsf{P}',\mathsf{D}}$-automaton $\mathbb{A}\restriction_{\mathsf{P}'}$ as the structure $\langle A, a_I, \Delta', \Omega \rangle$ given by

$$\Delta'(a) \;:=\; \{(\pi \cap \mathsf{P}', B) \in K_{\mathsf{P}',\mathsf{D}}(A) \mid (\pi, B) \in \Delta(a)\}.$$

◁

In words, Δ' is like Δ but keeps all options open for the proposition letters not in P'.

As an immediate corollary of the following proposition, the class of recognizable languages is closed under taking projections (modulo bisimulation).

Proposition 6.25 *Let $\mathsf{P}' \subseteq \mathsf{P}$ be two sets of proposition letters, and let $\mathbb{A}$ be some $K_{\mathsf{P},\mathsf{D}}$-automaton. Then*

1. *If $\mathbb{A}$ accepts some pointed Kripke model $(\mathbb{S}, s)$ over P, then $\mathbb{A}\restriction_{\mathsf{P}'}$ accepts $(\mathbb{S}\restriction_{\mathsf{P}'}, s)$.*
2. *If $\mathbb{A}\restriction_{\mathsf{P}'}$ accepts some Kripke model $(\mathbb{M}, m)$ over P', then $\mathbb{A}$ accepts some P-model $(\mathbb{S}, s)$ such that $\mathbb{M}, m \leftrightarrows_{\mathsf{P}'} \mathbb{S}\restriction_{\mathsf{P}'}, s$.*

Proof.

▶ `Proof details to be added.`

QED

Finally, we summarize our findings.

Theorem 6.26 *For any Kripke functor K, the class of recognizable K-languages is closed under taking unions, intersections, complementation, and projections modulo bisimulation.*

9 Results on the modal μ-calculus

In this chapter we gather some of the most important results concerning the modal μ-calculus.

9.1 Tree model property

Given the game-theoretic characterization of the semantics, it is rather straightforward to prove that formulas of the modal μ-calculus are bisimulation invariant. From this it is immediate that the modal mu-calculus has the tree model property. But in fact, we can use the game semantics to do better than this, proving that every satisfiable modal fixpoint formula is satisfied in a tree of which the branching degree is bounded by the size of the formula.

Theorem 9.1 (Bisimulation Invariance) *Let ξ be a modal fixpoint formula with $FV(\xi) \subseteq \mathsf{P}$, and let $\mathbb{S}$ and $\mathbb{S}'$ be two labelled transition systems with points s and s', respectively. If $\mathbb{S}, s \leftrightarrows_{\mathsf{P}} \mathbb{S}', s'$, then*

$$\mathbb{S}, s \Vdash \xi \text{ iff } \mathbb{S}', s' \Vdash \xi.$$

Proof. Assume that $s \leftrightarrows_{\mathsf{P}} s'$ and that $\mathbb{S}, s \Vdash \xi$, with $FV(\xi) \subseteq \mathsf{P}$. We will show that $\mathbb{S}', s' \Vdash \xi$. By the Adequacy Theorem we may assume that $\exists$ has a history free winning strategy f in the evaluation game $\mathcal{E} := \mathcal{E}(\xi, \mathbb{S})$ initialized at (ξ, s), and by the same theorem it suffices to provide her with a winning strategy in the game $\mathcal{E}' := \mathcal{E}(\xi, \mathbb{S}')@(\xi, s')$. She obtains her strategy f' in $\mathcal{E}'$ from playing a *shadow match* of $\mathcal{E}$, using the bisimilarity relation to guide her choices.

To see how this works, let's simply start with comparing the initial position (ξ, s') of $\mathcal{E}'$ with its counterpart (ξ, s) of $\mathcal{E}$. (From now on we will write $s \leftrightarrows s'$ instead of $s \leftrightarrows_{\mathsf{P}} s'$).

In case ξ is an atomic formula, then it is easy to see that both (ξ, s) and (ξ, s') are final positions. Also, since f is assumed to be winning, ξ must be true at s, and so it must hold at s' as well. Hence, $\exists$ wins the match.

If ξ is not atomic, we distinguish cases. First suppose that $\xi = \xi_1 \vee \xi_2$. If f tells $\exists$ to choose disjunct ξ_i at (ξ, s), then she chooses the same disjunct ξ_i at position (ξ, s'). If $\xi = \xi_1 \wedge \xi_2$, it is $\forall$ who moves. Suppose in $\mathcal{E}'$ he chooses ξ_i, making (ξ_i, s') the next position. We now consider in $\mathcal{E}$ the same move of $\forall$, so that the next position in the shadow match is (ξ_i, s).

A third possibility is that $\xi = \Diamond\psi$. In order to make her move at (ξ, s'), $\exists$ first looks at (ξ, s). Since f is a winning strategy, it indeed picks a successor t of s. Then because $s \leftrightarrows s'$, there is a successor t' of s' such that $t \leftrightarrows t'$. This t' is $\exists$'s move in $\mathcal{E}$, so that (ψ, t) and (ψ, t') are the next positions in $\mathcal{E}$ and $\mathcal{E}'$, respectively.

Finally, if $\xi = \Box\psi$, we are dealing again with positions for $\forall$. Suppose in $\mathcal{E}'$ he chooses the successor t' of s', so that the next position is (ψ, t'). (In case s' has no successors, $\forall$ immediately looses, so that there is nothing left to prove.) Now again we turn to the shadow match; by bisimilarity of s and s' there is a successor t of s such that $t \leftrightarrows t'$. So we may assume that $\forall$ moves the game token of $\mathcal{E}$ to position (ψ, t).

The crucial observation is that if $\exists$ does not win immediately, then at least she can guarantee that the next positions in $\mathcal{E}$ and $\mathcal{E}'$ are of the form (φ, u) and (φ, u') respectively, with $u \leftrightarrows u'$, and such that the move in $\mathcal{E}$ is consistent with f.

Continuing in this fashion, $\exists$ is able to maintain the condition (*) that for any match

$$\beta' = (\varphi_0, s'_0)(\varphi_1, s'_1)\dots(\varphi_n, s'_n)$$

played thus far, there is a shadow match

$$\beta = (\varphi_0, s_0)(\varphi_1, s_1)\dots(\varphi_n, s_n)$$

in $\mathcal{E}$ which is consistent with f, and such that $Z : s_i \leftrightarrows s'_i$ for all $i \leq n$.

It is not hard to see why this suffices to prove the theorem; for infinite matches, the key observation is that the sequence of formulas in the $\mathcal{E}'$-match and its $\mathcal{E}$-shadow are exactly the same. QED

As an immediate corollary, we obtain the tree model property for the modal μ-calculus.

Corollary 9.2 (Tree Model Property) *Let ξ be a modal fixpoint formula. If ξ is satisfiable, then it is satisfiable at the root of a tree model.*

Proof. For simplicity, we confine ourselves to the basic modal language. Suppose that ξ is satisfiable at state s of the Kripke model $\mathbb{S}$.

Define the *unravelling* $\vec{\mathbb{S}}_s$ of $\mathbb{S}$ at s as the following model. Its states are the *finite paths* through $\mathbb{S}$ that start at s. We define the (coalgebraic) structure $\vec{\sigma}$ on this set as follows:

$$\vec{\sigma}(s_0 \cdots s_k) := (\sigma_V(s_k), \{s_0 \cdots s_k s_{k+1} \mid s_{k+1} \in \sigma(s_k)\}).$$

It is left for the reader to verify that $\vec{\mathbb{S}}_s, s \leftrightarrows \mathbb{S}, s$. From this it follows by bisimulation invariance that $\vec{\mathbb{S}}, s \Vdash \xi$, and since it is clear that $\vec{\mathbb{S}}_s$ is a tree with root s, the result is immediate. QED

For the next theorem, recall that the size of a formula is simply defined as its length, that is, the number of symbols occurring in it.

Theorem 9.3 (Bounded Tree Model Property) *Let ξ be a modal fixpoint formula. If ξ is satisfiable, then it is satisfiable at the root of a tree, of which the branching degree is bounded by the size $|\xi|$ of the formula.*

Proof. Suppose that ξ is satisfiable. By the Bisimulation Invariance Theorem it follows that ξ is satisfiable at the root r of some tree model $\mathbb{T} = \langle T, R, V\rangle$. So, by the Adequacy Theorem, $\exists$ has a winning strategy f in the game $\mathcal{E} := \mathcal{E}(\xi, \mathbb{T})$ starting at position (ξ, r). And using the History-Free Determinacy of parity games, we may assume that this strategy is positional, that is, only depends on the current position in the game.

We will prune the tree $\mathbb{T}$, keeping only the nodes that $\exists$ needs in order to win the match. Formally, define subsets $(T_n)_{n\in\omega}$ as follows:

$$\begin{array}{lcl} T_0 & := & \{r\}, \\ T_{n+1} & := & T_n \cup \{s \mid (\varphi, s) = f(\Diamond\varphi, t) \text{ for some } t \in T_n \text{ and } \Diamond\varphi \trianglelefteq \xi\}, \\ T_\omega & := & \bigcup_{n\in\omega} T_n. \end{array}$$

Let $\mathbb{T}_\omega$ be the subtree of $\mathbb{T}$ based on T_ω (so $\mathbb{T}_\omega$ is not a generated submodel of $\mathbb{T}$). From the construction it is obvious that the branching degree of $\mathbb{T}_\omega$ is bounded by the length of ξ, because ξ has at most $|\xi|$ diamond subformulas.

We claim that $\mathbb{T}_\omega, r \Vdash \xi$. To see why this is so, let $\mathcal{E}' := \mathcal{E}(\xi, \mathbb{T}_\omega)$ be the evaluation game played on the pruned tree. It suffices to show that the strategy f', defined as the restriction of f to positions of the game $\mathcal{E}'$, is winning for $\exists$ in the game starting at (ξ, r). Consider an arbitrary $\mathcal{E}'$-match $\pi = (\xi, r)(\varphi_1, t_1)\ldots$ which is consistent with f'. The key observation of the proof is that π is also a match of $\mathcal{E}@(\xi, r)$, that is consistent with f. To see this, simply observe that all moves of $\forall$ in π could have been made in the game on $\mathbb{T}$ as well, whereas by construction, all f' moves of $\exists$ in $\mathcal{E}'$ are f moves in $\mathcal{E}$.

Now by assumption, f is a winning strategy for $\exists$ in $\mathcal{E}$, so she wins π in $\mathcal{E}$. But then π is winning as such, i.e., no matter whether we see it as a match in $\mathcal{E}$ or in $\mathcal{E}'$. In other words, π is also winning as an $\mathcal{E}'$-match. And since π was an arbitrary $\mathcal{E}'$ match starting at (ξ, r), this shows that f' is a winning strategy, as required. QED

9.2 Disjunctive normal form and decidability

As an immediate consequence of the fundamental result on Kripke automata (Theorem 6.17) it follows that every formula of the modal μ-calculus can be brought into so-called *disjunctive normal form*.

Definition 9.4 Given sets P of proposition letters, and D of atomic actions, respectively, the set $\mu\mathrm{CML}^-_{\mathsf{D}}(\mathsf{P})$ of *disjunctive* formulas is given by the following recursive definition:

$$\varphi \ ::= \ x \mid \bot \mid \top \mid \varphi \vee \varphi \mid \pi \bullet \Phi \mid \mu x.\varphi \mid \nu x.\varphi$$

where π denotes a subset of P, and $\Phi = \{\Phi_d \mid d \in \mathsf{D}\}$ a D-indexed collection of sets of disjunctive formulas, and x a variable *not* in P. $\lhd$

These formula are called disjunctive because the only admissible conjunctions are the special ones of the form $\pi \bullet \Phi$.

Theorem 9.5 *There is an effective algorithm that rewrites a modal fixpoint formula $\xi \in \mu\mathrm{PML}_{\mathsf{D}}(\mathsf{P})$ into an equivalent disjunctive formula ξ^d of length exponential in $|\xi|$.*

Proof. Immediate by Theorem 6.17. QED

Theorem 9.6 *There is an algorithm that decides in linear time whether a given disjunctive formula ξ is satisfiable or not.*

Proof. It is easy to see that the proof of this proposition is a direct consequence of the following observations:

1. $\top$ is satisfiable;
2. $\bot$ is not satisfiable;
3. $\varphi_1 \vee \varphi_2$ is satisfiable iff φ_1 or φ_2 is satisfiable;
4. $\pi \bullet \Phi$ is satisfiable iff both π and each $\varphi \in \bigcup_{d \in \mathsf{D}} \Phi_d$ is satisfiable;
5. $\mu x.\varphi$ is satisfiable iff $\varphi[\bot/x]$ is satisfiable;
6. $\nu x.\varphi$ is satisfiable iff $\varphi[\top/x]$ is satisfiable.

The proof of these claims is left as an exercise for the reader. QED

Decidability of the satisfiability problem for modal fixpoint formulas is then an immediate consequence of the previous two results.

Theorem 9.7 *There is an algorithm that decides in exponential time whether a given modal fixpoint formula ξ is satisfiable or not.*

9.3 Small model property

In this section we will show that any satisfiable modal fixpoint formula can in fact already be satisfied in a model of size at most exponential in the size of the formula. Given the Theorems 6.11 and Theorem 6.17, with any modal fixpoint formula ξ we may associate an equivalent *non-deterministic* Kripke automaton $\mathbb{A}$ of size exponential in ξ.

For convenience here we will denote the transition function of nondeterministic Kripke automaton $\mathbb{A} = \langle A, a_I, \Delta, \Omega \rangle$ as a map $\Delta : A \to \wp_\exists KA$, cf. Remark 6.10.

Definition 9.8 Let $\mathbb{A} = \langle A, a_I, \Delta, \Omega \rangle$ be a nondeterministic Kripke automaton, and let $\alpha : A \to KA$ be such that $\alpha(a) \in \Delta(a)$ for all $a \in A$. Then we say that the pointed Kripke model $\langle A, \alpha, a_I \rangle$ is a *realization* of $\mathbb{A}$. $\lhd$

Theorem 9.9 *Let $\mathbb{A} = \langle A, a_0, \Delta, \Omega \rangle$ be a nondeterministic Kripke automaton. If $\mathbb{A}$ accepts some pointed Kripke model, then it accepts one that is a realization of $\mathbb{A}$.*

Proof. Fix the automaton $\mathbb{A} = \langle A, a_0, \Delta, \Omega \rangle$. For simplicity we assume that we are dealing with a Kripke functor for the basic modal language.

Define the following *nonemptiness game* $\mathcal{N}(\mathbb{A})$. $\mathcal{N}(\mathbb{A})$ is a board game with board $A \cup KA \cup \wp A$. Its rules are given by the Table 11 below. The winning conditions for finite matches are standard; the winner of an infinite match is determined by the sequence of parities of states of $\mathbb{A}$ occurring as positions during the match.

Position	Player	Admissible moves
$a \in A$	$\exists$	$\Delta(a)$
$(\pi, B) \in KA$	-	$\{B\}$
$B \in \wp(A)$	$\forall$	B

Table 11: Nonemptiness game for nondeterministic Kripke automata

Intuitively, this game corresponds to the 'projection' on $\mathbb{A}$ of acceptance games associated with $\mathbb{A}$. We will make this intuition more precise in the following two claims, from which the theorem follows immediately.

Claim 1 If $\mathbb{A}$ accepts some pointed Kripke model, then a_0 is a winning position for $\exists$ in $\mathcal{N}(\mathbb{A})$.

For a proof of this claim, assume that $\mathbb{A}$ accepts the pointed Kripke model $(\mathbb{S}, s_0)$. Then $\exists$ has a winning strategy f in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ initialized at (a_0, s_0). Without loss of generality we may assume that this strategy leaves $\forall$ as little options as possible. In particular, at a position of the form $((\pi, B), s) \in KA \times S$, we may assume that she chooses a *minimal* relation $Z \subseteq A \times S$ such that $((\pi, B), \sigma(s)) \in \overline{K}(Z)$. So for instance, we may assume that Z only contains pairs $(b, t) \in B \times \sigma_R(s)$. Observe that it then follows from $((\pi, B), Z) \in \overline{K}(Z)$, that B must be the *domain* of Z, and $\sigma_R(s)$ the *range*.

She will use this strategy to win the game $\mathcal{N}(\mathbb{A})$ initialized at a_0. More precisely, in the latter game she will maintain the condition that with every partial match

$$a_0 \alpha_0 B_0 \dots a_n \alpha_i B_n a_{n+1}$$

in $\mathcal{N}(\mathbb{A})$ she can associate a shadow match

$$(a_0, s_0)(\alpha_0, s_0) Z_0 \dots (a_n, s_n)(\alpha_n, s_n) Z_n (a_{n+1}, s_{n+1})$$

in $\mathcal{A}(\mathbb{A}, \mathbb{S})$ such that $B_i = \mathrm{dom}(Z_i)$ for each i, and in which she plays her (winning) strategy f. The latter means that $\alpha_i = f(a_i, s_i)$ and $Z_i = f(\alpha_i, s_i)$, for each i.

It suffices to show inductively that she can prolong this condition for one more round. For this purpose, at a position a_i in $\mathcal{N}(\mathbb{A})$ she looks at the corresponding $\mathcal{A}(\mathbb{A}, \mathbb{S})$-position (a_i, s_i); if there her strategy f tells her to move to (β, s_i), her choice in $\mathcal{N}(\mathbb{A})$ will be β. Write $\beta = (\pi, B)$, then the next position in $\mathcal{N}(\mathbb{A})$ is B. In $\mathcal{A}(\mathbb{A}, \mathbb{S})$, at position (β, s_i), f tells $\exists$ to choose some local bisimulation $Z \subseteq A \times S$ with $((\pi, B), \sigma(s)) \in \overline{K}(Z)$. By our assumption on $\exists$'s winning strategy f, B is the domain of Z. Finally, if $\forall$ at position B in $\mathcal{N}(\mathbb{A})$ chooses a new position $b \in B$, in the shadow match, $\exists$ lets $\forall$ play some position $(b, t) \in Z$. This is a legitimate move, because Z is *full* on B and $\sigma_R(s)$, so that in particular for every $b \in B$ there is a $t \in \sigma_R(s)$ with $(b, t) \in Z$.

Note that maintaining the above mentioned condition, she effectively plays some strategy that we now prove to be winning. Recall that by assumption, f is winning.

▶ `Leave the case of finite matches to the reader`

In the case of an infinite match, the sequence $a_0 a_1 a_2 \ldots$ satisfies the parity condition. But then the $\mathcal{N}(\mathbb{A})$-match is won by $\exists$ as well. This finishes the proof of the claim.

Now the point of introducing this nonemptiness game is that it is a *parity* game, and hence, it satisfies History Free Determinacy. That is, we may assume the existence of a history free strategy for $\exists$ that is winning when played from any position in $\mathrm{Win}_\exists$. Given the nature of $\exists$'s positions, this strategy will be a map $\alpha : A \to KA$. That is, her winning strategy *uniquely determines* a pointed Kripke model $\mathbb{A}_\alpha$ living inside $\mathbb{A}$.

Claim 2 If a_0 is a winning position for $\exists$ in $\mathcal{N}(\mathbb{A})$, then $\mathbb{A}$ accepts $\mathbb{A}_\alpha$.

For a *proof* of this claim it suffices to prove that the following strategy for $\exists$ guarantees that she wins the acceptance game associated with $\mathbb{A}$ and $\mathbb{A}_\alpha$:

- at a position of the form (a, a), choose $(\alpha(a), a)$;
- at a position of the form $(\alpha(a), a)$, with $\alpha(a) = (\pi, B)$, choose the set $Id_B = \{(b, b) \mid b \in B\}$ as a candidate local bisimulation;
- at other positions, choose randomly.

It is not hard to see that playing this strategy, $\exists$ guarantees that only basic positions (a, a) with $a \in \mathrm{Win}_\exists(\mathcal{N}(\mathbb{A}))$ are played. In fact, it is easy to prove as well that the resulting $\mathcal{A}(\mathbb{A}, \mathbb{A}_\alpha)$-match is of the form

$$(a_0, a_0) \ldots (a_i, a_i)(\alpha(a_i), a_i) Id_{B_i} (a_{i+1}, a_{i+1})(\alpha(a_{i+1}), a_{i+1}) Id_{B_{i+1}} \ldots$$

such that the corresponding match of $\mathcal{N}(\mathbb{A})$ is of the form

$$a_0 \dots a_i \alpha(a_i) B_i a_{i+1} \alpha(a_{i+1}) B_{i+1} \dots$$

(Here every $\alpha(a_i)$ is of the form (π_i, B_i).)

From this and the assumption that $a_0 \in \mathrm{Win}_\exists(\mathcal{N}(\mathbb{A}))$, it is immediate that the $\mathcal{A}(\mathbb{A}, \mathbb{A}_\alpha)$-match is won by $\exists$. Thus the strategy described above is a winning strategy, which suffices to prove the claim, and hence, the theorem. QED

Corollary 9.10 *Let $\xi \in \mu\mathrm{ML}(\mathsf{P})$ be a modal fixpoint formula. Then ξ is satisfiable iff ξ is satisfiable in a model of size at most $2^{|\xi|}$.*

Proof.

▶ `Immediate by ...`

QED

9.4 Expressive completeness

Theorem 9.11 *Let φ be a monadic second order property of LTSs which is bisimulation invariant. Then there is a μPML-formula φ' which is equivalent to φ.*

9.5 Axiomatization

Definition 9.12 Let $\mathbf{K}\mu$ be the logic obtained by adding the following axiom scheme

(prefix) $\varphi[\mu x.\varphi/x] \to \mu x.\varphi$

and derivation rule:

(min) from $\vdash \varphi[\psi/x] \to \psi$ infer $\vdash \mu x.\varphi \to \psi$.

to the basic modal logic **K**. ◁

Clearly, (prefix) expresses that $\mu x.\varphi$ is a prefixpoint of the formula $\varphi(x)$, and (min) says that $\mu x.\varphi$ is in fact below any prefixpoint of $\varphi(x)$.

Theorem 9.13 $\mathbf{K}\mu$ *is sound and complete with respect to the standard semantics.*

9.6 Uniform interpolation

Definition 9.14 Given two modal fixpoint formulas φ and ψ, we say that ψ is a *(local) consequence* of φ, notation: $\varphi \models \psi$, if $\mathbb{S}, s \Vdash \varphi$ implies $\mathbb{S}, s \Vdash \psi$, for every pointed Kripke model $(\mathbb{S}, s)$. $\lhd$

Recall that a formalism has the *interpolation property* if we can find an *interpolant* for every pair of formulas φ and ψ such that $\varphi \models \psi$. This interpolant is a formula θ such that $\varphi \models \theta$ and $\theta \models \psi$; but most importantly, the requirement on θ is that it may only use symbols that occur both in φ and ψ.

▶ `why this is an important property`

As we will see now, the modal μ-calculus has *uniform* interpolation. This is a very strong version of interpolation in which the interpolant θ does not depend on the shape of ψ, but only on the language of ψ. More precisely, we define the following.

Definition 9.15 Let φ be a modal fixpoint formula, and $\mathsf{Q} \subseteq FV(\varphi)$. Then a *uniform interpolant* of φ with respect to Q is a formula θ with $FV(\theta) \subseteq \mathsf{Q}$, such that for all formulas ψ with $FV(\psi) \cap FV(\varphi) \subseteq \mathsf{Q}$:

$$\varphi \models \psi \text{ iff } \theta \models \psi. \tag{14}$$

$\lhd$

Remark 9.16 Instead of (14) we could have required

$$\varphi \models \psi \text{ iff } \varphi \models \theta \text{ and } \theta \models \psi, \tag{15}$$

which perhaps shows more clearly that θ is indeed an interpolant.

These two definitions are in fact equivalent. The key observation to see this is that (14) implies that from $\theta \models \theta$ it follows that $\varphi \models \theta$. $\lhd$

Theorem 9.17 (Uniform Interpolation) *Every modal fixpoint formula has a uniform interpolant.*

The proof consists of showing that the modal μ-calculus can express *bisimulation quantifiers*.

Proof. Fix the formula φ and the set Q, and define $\mathsf{P} := FV(\varphi)$ and $\mathsf{R} := \mathsf{P} \setminus \mathsf{Q}$.

By the equivalence of modal fixpoint formulas and nondeterministic Kripke automata, it follows from Proposition 6.25 that the modal μ-calculus can express *bisimulation quantifiers*. That is, given φ and Q, there is a formula $I_{\mathsf{Q}}(\varphi)$ with $FV(I_{\mathsf{Q}}(\varphi)) \subseteq \mathsf{Q}$ such that, for all pointed Q-models $(\mathbb{M}, s)$:

$$\mathbb{M}, s \Vdash I_{\mathsf{Q}}(\varphi) \text{ iff } \mathbb{S}, s \Vdash \varphi \text{ for some } \mathsf{P}\text{-model } \mathbb{S} \text{ with } \mathbb{S}{\restriction_{\mathsf{Q}}} \leftrightarrows_{\mathsf{Q}} \mathbb{M}.$$

We leave the fairly straightforward (but not completely trivial) task of verifying that this $I_{\mathsf{Q}}(\varphi)$ is a uniform interpolant of φ with respect to Q as an exercise for the reader. QED

9.7 Model checking

▶ `Details to be supplied`

A Basic fixpoint theory

Orders and lattices

Definition A.1 A *partial order* is a structure $\mathbb{P} = \langle P, \leq \rangle$ such that $\leq$ is a reflexive, transitive and antisymmetric relation on P.

Given a partial order $\mathbb{P}$, an element $p \in P$ is an *upper bound* (*lower bound, respectively*) of a set $X \subseteq P$ if $p \geq x$ for all $x \in X$ ($p \leq x$ for all $x \in X$, respectively). If the set of upper bounds of X has a minimum, this element is called the *least upper bound*, *supremum*, or *join* of X, notation: $\bigvee X$. Dually, the *greatest lower bound*, *infimum*, or *meet* of X, if existing, is denoted as $\bigwedge X$.

A partial order $\mathbb{P}$ is called a *lattice* if every two-element subset of P has both an infimum and a supremum; in this case, the notation is as follows: $p \wedge q := \bigwedge\{p, q\}$, $p \vee q := \bigvee\{p, q\}$. Such a lattice is *bounded* if it has a minimum $\bot$ and a maximum $\top$.

A partial order $\mathbb{P}$ is called a *complete lattice* if every subset of P has both an infimum and a supremum. In this case we abbreviate $\bot := \bigvee \varnothing$ and $\top := \bigwedge \varnothing$; these are the smallest and largest elements of $\mathbb{C}$, respectively. A complete lattice will usually be denoted as a structure $\mathbb{C} = \langle C, \bigvee, \bigwedge \rangle$. $\lhd$

Definition A.2 Given a family $\{\mathbb{P}_i \mid i \in I\}$ of partial orders, we define the *product* order $\prod_{i \in I} \mathbb{P}_i$ as the structure $\langle \prod_{i \in I} P_i, \leq \rangle$ where $\prod_{i \in I} P_i$ denotes the cartesian product of the family $\{P_i \mid i \in I\}$, and $\leq$ is given by $\pi \leq \pi'$ iff $\pi(i) \leq_i \pi'(i)$ for all $i \in I$. $\lhd$

It is not difficult to see that the product of a family of (complete) lattices is again a (complete) lattice, with meets and joins given pointwise. For instance, given a family $\{\mathbb{C}_i \mid i \in I\}$ of complete lattices, and a subset $\Gamma \subseteq \prod_{i \in I} C_i$, it is easy to see that Γ has a least upper bound $\bigvee \Gamma$ given by

$$(\bigvee \Gamma)(i) = \bigvee\{\gamma(i) \mid \gamma \in \Gamma\},$$

where the join on the right hand side is taken in $\mathbb{C}_i$.

Definition A.3 Let $\mathbb{P}$ and $\mathbb{P}'$ be two partial orders and let $f : P \to P'$ be some map. Then f is called *monotone* if $f(x) \leq' f(y)$ whenever $x \leq y$, and *antitone* if $f(x) \geq' f(y)$ whenever $x \leq y$. $\lhd$

Fixpoints

Definition A.4 Let $\mathbb{P} = \langle P, \leq \rangle$ be some partial order, and let $f : P \to P$ be some map. Then an element $p \in P$ is called a *prefixpoint* of f if $f(p) \leq p$, a *postfixpoint* of f if $f(p) \geq p$, and a *fixpoint* if $f(p) = p$. The sets of prefixpoints, postfixpoints, and fixpoints of f are denoted respectively as $\mathrm{PRE}(f)$, $\mathrm{POS}(f)$ and $\mathrm{FIX}(f)$.

In case the set of fixpoints of f has a least, respectively greatest member, this element is denoted as $\mathrm{LFP}.f$, ($\mathrm{GFP}.f$, respectively). $\lhd$

The following theorem is a celebrated result in fixpoint theory.

Theorem A.5 (Knaster-Tarski) *Let $\mathbb{C} = \langle C, \bigvee, \bigwedge \rangle$ be a complete lattice, and let $f : P \to P$ be monotone. Then f has both a least and a greatest fixpoint, and these are given as*

$$\begin{aligned} \mathrm{LFP}.f &= \bigwedge \mathrm{PRE}(f), \\ \mathrm{GFP}.f &= \bigvee \mathrm{POS}(f). \end{aligned}$$

Proof. We will only prove the result for the least fixpoint, the proof for the greatest fixpoint is completely analogous.

Define $q := \bigwedge \mathrm{PRE}(f)$, then we have that $q \leq x$ for all prefixpoints x of f. From this it follows that $f(q) \leq f(x)$ for all such x, and hence by definition of prefixpoints, $f(q) \leq x$ for all $x \in \mathrm{PRE}(f)$, so that $f(q)$ is a lower bound of the set $\mathrm{PRE}(f)$. Hence, by definition of q as the *greatest* such lower bound, we find $f(q) \leq q$. In other words: q itself is a prefixpoint of f.

It now suffices to prove that $q \leq f(q)$, and for this we may show that $f(q)$ is a prefixpoint of f as well, since q is by definition a lower bound of the set of prefixpoints. But in fact, we may show that $f(y)$ is a prefixpoint of f for *every* prefixpoint y of f — this immediately follows by monotonicity of f. QED

Another way to obtain least and greatest fixpoint is to *approximate* them from below, respectively, above.

Definition A.6 Let $\mathbb{C} = \langle C, \bigvee, \bigwedge \rangle$ be a complete lattice, and let $f : C \to C$ be some map. Then by ordinal induction we define the following maps on C:

$$\begin{aligned} f_\mu^0(c) &:= c, \\ f_\mu^{\alpha+1}(c) &:= f(f_\mu^\alpha(c)) \\ f_\mu^\lambda(c) &:= \bigvee_{\alpha<\lambda} f_\mu^\alpha(c), \end{aligned}$$

where λ denotes an arbitrary limit ordinal. Dually, we put

$$\begin{aligned} f_\nu^0(c) &:= c, \\ f_\nu^{\alpha+1}(c) &:= f(f_\nu^\alpha(c)), \\ f_\nu^\lambda(c) &:= \bigwedge_{\alpha<\lambda} f_\nu^\alpha(c), \end{aligned}$$

◁

Proposition A.7 *Let $\mathbb{C} = \langle C, \bigvee, \bigwedge \rangle$ be a complete lattice, and let $f : C \to C$ be monotone. Then f is inductive, that is, $f_\mu^\alpha(\bot) \leq f_\mu^\beta(\bot)$ for all ordinals α and β such that $\alpha < \beta$.*

Proof. Proof to be supplied QED

Corollary A.8 *Let $\mathbb{C} = \langle C, \bigvee, \bigwedge \rangle$ be a complete lattice, and let $f : C \to C$ be monotone. Then there is some $\alpha \leq |C|$ such that* LFP.$f = f_\mu^\alpha(\bot)$.

Proof. By inductiveness (Proposition A.7) there must be some ordinal $\alpha \leq |C|$ such that $f_\mu^\alpha(\bot) = f_\mu^{\alpha+1}(\bot)$. From this it follows that $f_\mu^\alpha(\bot)$ is a fixpoint of f. To show that it is the *smallest* fixpoint, one may prove that $f_\mu^\beta(\bot) \leq$ LFP.f for every ordinal β. This follows from a straightforward ordinal induction. QED

Definition A.9 Let $\mathbb{C} = \langle C, \bigvee, \bigwedge \rangle$ be a complete lattice, and let $f : C \to C$ be monotone. The least ordinal α such that $f_\mu^\alpha(\bot) = f_\mu^{\alpha+1}(\bot)$ is called the *closure ordinal* of f. $\lhd$

Multidimensional fixpoints

Suppose that we are given a finite family $\{\mathbb{C}_1, \ldots, \mathbb{C}_n\}$ of complete lattices, and put $\mathbb{C} = \prod_{1 \leq i \leq n} \mathbb{C}_i$. Given a finite family of monotone maps $f_1, \ldots, f_n$ with $f_i : C \to C_i$, we may define the map $f : C \to C$ given by $f(c) = (f_1(c), \ldots, f_n(c))$. It is easy to prove that this f is monotone, so that by completeness of $\mathbb{C}$ it has a least and a greatest fixpoint. An obvious question is whether one may express these multi-dimensional fixpoints in terms of one-dimensional fixpoints of maps that one may associate with $f_1, \ldots, f_n$.

It will be convenient to introduce some notation. Given a map $g : C \to C_i$ and an $n-1$-tuple $\bar{x} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$, we let $g_{\bar{x}} : C_i \to C_i$ denote the map given by

$$g_{\bar{x}}(x_i) := g(x_1, x_2, \ldots, x_n).$$

The least and greatest fixpoints of this operation will be denoted as $\mu x_i.g(x_1, x_2, \ldots, x_n)$ and $\nu x_i.g(x_1, x_2, \ldots, x_n)$, respectively.

Furthermore, in this context we will also use vector notation, for instance writing

$$\mu \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} . \begin{pmatrix} f_1(x_1, \ldots, x_n) \\ f_2(x_1, \ldots, x_n) \\ \vdots \\ f_2(x_1, \ldots, x_n) \end{pmatrix}$$

for LFP.f.

The basic observation facilitating the computation of multidimensional fixpoints is the following.

Proposition A.10 *Let $\mathbb{D}_1$ and $\mathbb{D}_2$ be two complete lattices, and let $f_i : D_1 \times D_2 \to D_i$ for $i = 1, 2$ be monotone maps. Then*

$$\eta \begin{pmatrix} x \\ y \end{pmatrix} . \begin{pmatrix} f_1(x,y) \\ f_2(x,y) \end{pmatrix} = \begin{pmatrix} \eta x.f_1(x, \eta y.f_2(x,y)) \\ \eta y.f_2(\eta x.f_1(x,y), y) \end{pmatrix}$$

where η uniformly denotes either μ or ν.

Proof.

▶ `To be supplied`

QED

Using induction on the dimension, Proposition A.10 allows us to compute the least and greatest fixpoints of any monotone map f on a complete finite product lattice. The correctness of this *elimination method*, which is reminiscent of Gauss elimination in linear algebra, is a direct consequence of Proposition A.10.

To see how it works, suppose that we are dealing with lattices $\mathbb{C}_1, \ldots, \mathbb{C}_{n+1}, \mathbb{C}$ and maps $f_1, \ldots, f_{n+1}, f$, just as described above, and that we want to compute $\eta\vec{x}.f$, that is, find the elements $a_1, \ldots, a_{n+1}$ such that

$$\begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{n+1} \end{pmatrix} = \eta \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n+1} \end{pmatrix} . \begin{pmatrix} f_1(x_1, \ldots, x_n, x_{n+1}) \\ f_2(x_1, \ldots, x_n, x_{n+1}) \\ \vdots \\ f_{n+1}(x_1, \ldots, x_n, x_{n+1}) \end{pmatrix}$$

Using Proposition A.10, with $\mathbb{D}_1 = \mathbb{C}_1 \times \cdots \times \mathbb{C}_n$, and $\mathbb{D}_2 = \mathbb{C}_{n+1}$, we may first compute

$$g_{n+1}(x_1, \ldots, x_n) := \eta x_{n+1}.f_{n+1}(x_1, \ldots, x_n).$$

We then *insert* this in the remaining equations, and recursively obtain a solution

$$\begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} = \eta \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} . \begin{pmatrix} f_1(x_1, \ldots, x_n, g_{n+1}(x_1, \ldots, x_n)) \\ f_2(x_1, \ldots, x_n, g_{n+1}(x_1, \ldots, x_n)) \\ \vdots \\ f_n(x_1, \ldots, x_n, g_{n+1}(x_1, \ldots, x_n)) \end{pmatrix}$$

Finally, we may compute $a_{n+1} := g_{n+1}(a_1, \ldots, a_n)$.

Observe that in case $\mathbb{C}_i = \mathbb{C}_j$ for all i, j and the operations f_i are all term definable in some formal algebraic fixpoint language, then each the components a_i of the extremal fixpoints of f can also be expressed in this language.

Boolean algebras

In the case of monotone maps on complete Boolean algebras, least and greatest fixed points become interdefinable, using the notion of (Boolean) *duals* of maps.

Definition A.11 A *complete* Boolean algebra is a structure $\mathbb{B} = \langle B, \bigvee, \bigwedge, - \rangle$ such that $\langle B, \bigvee, \bigwedge \rangle$ is a complete lattice, and $- : B \to B$ is an antitone map such that $x \wedge -x = \bot$ and $x \vee -x = \top$ for all $x \in B$. $\lhd$

It is not too hard to verify that in a complete Boolean algebra $\mathbb{B} = \langle B, \bigvee, \bigwedge, - \rangle$, it holds that $-\bigvee X = \bigwedge\{-x \mid x \in X\}$ and $-\bigwedge X = \bigvee\{-x \mid x \in X\}$.

Definition A.12 Let $\mathbb{B} = \langle B, \bigvee, \bigwedge, - \rangle$ be a complete Boolean algebra, and let $f : B \to B$ be some map. Then the *(Boolean) dual map* of f is defined as the map $\tilde{f} : B \to B$ given by

$$\tilde{f}(b) := -f(-x).$$

$\lhd$

Proposition A.13 *Let $\mathbb{B} = \langle B, \bigvee, \bigwedge, - \rangle$ be a complete Boolean algebra, and let $f : B \to B$ be monotone. Then $\tilde{f}$ is monotone as well, $\tilde{\tilde{f}} = f$, and*

$$\begin{aligned} \mathrm{LFP}.\tilde{f} &= -\mathrm{GFP}.f, \\ \mathrm{GFP}.\tilde{f} &= -\mathrm{LFP}.f. \end{aligned}$$

Proof. We only prove that $\mathrm{LFP}.\tilde{f} = -\mathrm{GFP}.f$, leaving the other parts of the proof as exercises to the reader.

First, note that by monotonicity of $\tilde{f}$, the Knaster-Tarski theorem gives that

$$\mathrm{LFP}.\tilde{f} = \bigwedge \mathrm{PRE}(\tilde{f}).$$

But as an easy consequence of the definitions, we have that

$$b \in \mathrm{PRE}(\tilde{f}) \iff -b \in \mathrm{POS}(f).$$

From this it follows that

$$\begin{aligned} \mathrm{LFP}.\tilde{f} &= \bigwedge\{-b \mid b \in \mathrm{POS}(f)\} \\ &= -\bigvee \mathrm{POS}(f) \\ &= -\mathrm{GFP}.f. \end{aligned}$$

QED

B Mathematical preliminaries

Sets, relations and functions We use standard notation for set-theoretic operations such as union, intersection, product, etc. The power set of a set S is denoted as $\wp(S)$ or $\wp S$, and we sometimes denote the relative complement operation as $\sim_S X := S \setminus X$.

Given a relation $R \subseteq A \times B$, we introduce the following notation. R^{-1} denotes the converse of R. For $X \subseteq A$, we put $R[X] := \{b \in B \mid (a,b) \in R \text{ for some } a \in X\}$; in case $X = \{s\}$ is a singleton, we write $R[s]$ instead of $R[\{s\}]$. For $Y \subseteq B$, we will write $\langle R \rangle Y$ rather than $R^{-1}[Y]$, while $[R]Y$ denotes the set $\{a \in A \mid b \in Y \text{ whenever } (a,b) \in R\}$. Note that $[R]Y = A \setminus exopR(B \setminus Y)$. For $R \subseteq S \times S$, R^* denotes the reflexive-transitive closure of R, and R^+ the transitive closure.

Let $f : A \to B$ be a function from A to B. Given a set $X \subseteq A$, we let $f[X] := \{f(a) \in B \mid a \in X\}$ denote the image of X under f, and given $Y \subseteq B$, $f^{-1}[Y] := \{a \in A \mid f(a) \in Y\}$ denotes the preimage of Y. In case f is a bijection, we let f^{-1} denote its inverse. The composition of two functions $f : A \to B$ and $g : B \to A$ is denoted as $g \circ f$ or gf, and the set of function from A to B will be denoted as either B^A or $A \to B$.

It is well-known that there is a bijective correspondence

$$(A \times B) \to C \;\cong\; A \to (B \to C).$$

With a function $f : A \times B \to C$, associate the map that, for each $a \in A$, yields the function $f_a : B \to C$ given by $f_a(b) := f(a,b)$.

Sequences, lists and streams Given a set Σ, we define Σ^* and Σ^ω as the set of *lists* or finite sequences over Σ, and the set of *streams* or ω-length sequences over Σ, respectively. We write $\Sigma^\infty := \Sigma^* \cup \Sigma^\omega$ for the set of all sequences over Σ.

We will write ε for the empty sequence, and use $\sqsubseteq$ for the initial segment relation between sequences, and $\sqsubset$ for the proper (i.e., irreflexive) version of this relation. For a nonempty sequence π, $\mathit{first}(\pi)$ denotes the first element of π. In the case that π is finite we write $\mathit{last}(\pi)$ for the last element of π.

References

[1] A. Arnold and D. Niwiński. *Rudiments of μ-calculus*. Number 146 in Studies in Logic. North-Holland, Elsevier, 2001.

[2] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Number 53 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.

[3] E.Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logic, and Infinite Games*. Number 2500 in *Lecture Notes in Computer Science*. Springer, 2002.

[4] C. Stirling. *Modal and Temporal Properties of Processes*. Texts in Computer Science. Springer-Verlag, 2001.