Lectures on the modal μ-calculus

Yde Venema*

Abstract

These notes give an introduction to the theory of the modal μ-calculus and other modal fixpoint logics.

*Institute for Logic, Language and Computation, University of Amsterdam, Science Park 107, NL–1098XG Amsterdam. E-mail: y.venema@uva.nl.

Contents

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. Such formalisms then have important applications in the specification and verification of software. For such purposes, 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 an exponential time satisfiability problem.

Modal Logic From the perspective of modal logic, the modal μ-calculus is a well-behaved extension of the basic 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 generally, 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 unknown 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. Reason enough to study it in more detail.

1 Basic 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. We focus on some aspects of modal logic that feature prominently in its extensions with fixpoint operators.

Convention 1.1 Throughout this text we let Prop be a countably infinite set of *propositional variables*, whose elements are usually denoted as $p, q, r, x, y, z, \ldots$, and we let D be a finite set of (atomic) *actions*, whose elements are usually denoted as $d, e, c, \ldots$. We will usually focus on a finite subset P of Prop, consisting of those propositional variables that occur freely in a particular formula. In practice we will often suppress explicit reference to Prop, P and D.

1.1 Basics

Structures

▶ `Introduce LTSs as process graphs`

Definition 1.2 A *(labelled) transition system*, *LTS*, or *Kripke model* of *type* (P, D) is a triple $\mathbb{S} = \langle S, V, R \rangle$ such that S is a set of objects called *states* or *points*, $V : \mathsf{P} \to \wp(S)$ 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 $R_d[s] := \{t \in S \mid (s, t) \in R_d\}$ are called *d-successors* of s. A transition system is called *image-finite* or *finitely branching* if $R_d[s]$ is finite, for every $d \in \mathsf{D}$ and $s \in S$.

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

Remark 1.3 It will occasionally be convenient to work with an alternative, *coalgebraic* presentation of transition systems. Intuitively, it should be clear that instead of having a valuation $V : \mathsf{P} \to \wp(S)$, telling us at which states each proposition letter is true, we could just as well have a *marking* $\sigma_V : S \to \wp(\mathsf{P})$ informing us which proposition letters are true at each state. Also, a binary relation R on a set S can be represented as a map $R[\cdot] : S \to \wp(S)$ mapping a state s to the collection $R[s]$ of its successors. In this line, a family $R = \{R_d \subseteq S \times S \mid d \in \mathsf{D}\}$ of accessibility relations can be seen as a map $\sigma_R : S \to \wp(S)^{\mathsf{D}}$, where $\wp(S)^{\mathsf{D}}$ denotes the set of maps from D to $\wp(S)$.

Combining these two maps into one single function, we see that a transition system $\mathbb{S} = \langle S, V, R \rangle$ of type (P, D) can be seen as a pair $\langle S, \sigma \rangle$, where $\sigma : S \to \wp(\mathsf{P}) \times \wp(S)^{\mathsf{D}}$ is the map given by $\sigma(s) := (\sigma_V(s), \sigma_R(s))$. $\lhd$

For future reference we define the notion of a *Kripke functor*.

Definition 1.4 Fix a set P of proposition letters and a set D of atomic actions. Given a set S, let $\mathsf{K}_{\mathsf{D},\mathsf{P}} S$ denote the set

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

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

A typical element of $\mathsf{K}_{\mathsf{D},\mathsf{P}}S$ 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 $d \in \mathsf{D}$.

When we take this perspective we will sometimes refer to Kripke models as $\mathsf{K}_{\mathsf{D},\mathsf{P}}S$-*coalgebras* or *Kripke coalgebras*. $\triangleleft$

Given this definition we may summarize Remark 1.3 by saying that any transition system can be presented as a pair $\mathbb{S} = \langle S, \sigma : S \to \mathsf{K}S\rangle$ where K is the Kripke functor associated with $\mathbb{S}$. In practice, we will usually write K rather than $\mathsf{K}_{\mathsf{D},\mathsf{P}}$.

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 1.5 The language $\mathtt{ML}_{\mathsf{D}}$ of *polymodal logic* in D is defined as follows:

$$\varphi \ ::= \ p \mid \overline{p} \mid \bot \mid \top \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \Diamond_d \varphi \mid \Box_d \varphi$$

where $p \in \mathsf{Prop}$, and $d \in \mathsf{D}$. Elements of $\mathtt{ML}_{\mathsf{D}}$ are called *(poly-)modal formulas*, or briefly, *formulas*. Formulas of the form $\bot$, $\top$, p or $\overline{p}$ are called *literals*. In case the set D is a singleton, we speak of the language $\mathtt{ML}$ of *basic modal logic* or *monomodal* logic; in this case we will denote the modal operators by $\Diamond$ and $\Box$, respectively.

Given a finite set P of propositional variables, we let $\mathtt{ML}_{\mathsf{D}}(\mathsf{P})$ denote the set of formulas in which only variables from P occur. $\triangleleft$

Often the sets P and D are implicitly understood, and suppressed in the notation. Generally it will suffice to treat examples, proofs, etc., from monomodal logic.

Remark 1.6 The *negation* $\sim\!\varphi$ of a formula φ can inductively be defined as follows:

$$\begin{array}{lcllcl}
\sim\bot & := & \top & \sim\top & := & \bot \\
\sim p & := & \overline{p} & \sim\overline{p} & := & p \\
\sim(\varphi \vee \psi) & := & \sim\varphi \wedge \sim\psi & \sim(\varphi \wedge \psi) & := & \sim\varphi \vee \sim\psi \\
\sim\Box_d \varphi & := & \Diamond_d \sim\varphi & \sim\Diamond_d \varphi & := & \Box_d \sim\varphi
\end{array}$$

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

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 $\varphi[\psi/p]$ denote the formula that we obtain by substituting all occurrences of p in φ by ψ.

Definition 1.7 We define the collection $Sfor(\xi)$ of subformulas of a modal formula ξ by the following induction on the complexity of ξ:

$$\begin{array}{lcll}
Sfor(\bot) & := & \{\bot\} & \\
Sfor(\top) & := & \{\top\} & \\
Sfor(p) & := & \{p\} & \\
Sfor(\overline{p}) & := & \{\overline{p}\} & \\
Sfor(\varphi \star \psi) & := & \{\varphi \star \psi\} \cup Sfor(\varphi) \cup Sfor(\psi) & \text{where } \star \in \{\vee, \wedge\} \\
Sfor(\heartsuit\varphi) & := & \{\heartsuit\varphi\} \cup Sfor(\varphi) & \text{where } \heartsuit \in \{\Diamond_d, \Box_d \mid d \in \mathsf{D}\}
\end{array}$$

We write $\varphi \trianglelefteq \psi$ to denote that φ is a subformula of ψ. The *size* of a formula ξ is defined as the number of its subformulas, $|\xi| := |Sfor(\xi)|$. $\triangleleft$

Semantics

The *relational* semantics of modal logic is well known. The basic idea is that the modal operators $\Diamond_d$ and $\Box_d$ are both interpreted using the *accessibility* relation R_d.

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

Definition 1.8 Let $\mathbb{S} = \langle S, \sigma \rangle$ be a transition system of type (P, D). Then the *satisfaction relation* $\Vdash$ between states of $\mathbb{S}$ and formulas of $\mathtt{ML}_{\mathsf{D}}(\mathsf{P})$ is defined by the following formula induction.

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

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 $\mathbb{S}$. $\triangleleft$

Alternatively (but equivalently), one may define the semantics of modal formulas directly in terms of this meaning function $[\![\varphi]\!]^{\mathbb{S}}$. This approach has some advantages in the context of fixpoint operators, since it brings out the role of the powerset algebra $\wp(S)$ more clearly.

Remark 1.9 Fix an LTS $\mathbb{S}$, then define $[\![\varphi]\!]^{\mathbb{S}}$ by induction on the complexity of φ:

$$\begin{array}{lcl}
[\![p]\!]^{\mathbb{S}} & = & V(p) \\
[\![\overline{p}]\!]^{\mathbb{S}} & = & S \setminus V(p) \\
[\![\bot]\!]^{\mathbb{S}} & = & \varnothing \\
[\![\top]\!]^{\mathbb{S}} & = & S \\
[\![\varphi \lor \psi]\!]^{\mathbb{S}} & = & [\![\varphi]\!]^{\mathbb{S}} \cup [\![\psi]\!]^{\mathbb{S}} \\
[\![\varphi \land \psi]\!]^{\mathbb{S}} & = & [\![\varphi]\!]^{\mathbb{S}} \cap [\![\psi]\!]^{\mathbb{S}} \\
[\![\Diamond_d \varphi]\!]^{\mathbb{S}} & = & \langle R_d \rangle [\![\varphi]\!]^{\mathbb{S}} \\
[\![\Box_d \varphi]\!]^{\mathbb{S}} & = & [R_d] [\![\varphi]\!]^{\mathbb{S}}
\end{array}$$

Here the operations $\langle R_d \rangle$ and $[R_d]$ on $\wp(S)$ are defined by putting

$$\begin{array}{lcl}
\langle R_d \rangle (X) & := & \{s \in S \mid R_d[s] \cap X \neq \varnothing\} \\
[R_d](X) & := & \{s \in S \mid R_d[s] \subseteq X\}.
\end{array}$$

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

Definition 1.10 Let s and s' be two states in the transition systems $\mathbb{S}$ and $\mathbb{S}'$ of type (P, D), respectively. Then we say that s and s' are *modally equivalent*, notation: $\mathbb{S}, s \equiv_{(\mathsf{P},\mathsf{D})} \mathbb{S}', s'$, if s and s' satisfy the same modal formulas, that is, $\mathbb{S}, s \Vdash \varphi$ iff $\mathbb{S}', s' \Vdash \varphi$, for all modal formulas $\varphi \in \mathtt{ML}_{\mathsf{D}}(\mathsf{P})$. $\lhd$

Flows, trees and streams

In some parts of these notes *deterministic* transition systems feature prominently.

Definition 1.11 A transition system $\mathbb{S} = \langle S, V, R \rangle$ is called *deterministic* if each $R_d[s]$ is a singleton. $\lhd$

Note that our definition of determinism does not allow $R_d = \varnothing$ for any point s. We first consider the monomodal case.

Definition 1.12 Let P be a set of proposition letters. A deterministic monomodal Kripke model for this language is called a *flow model for* P, or a $\wp(\mathsf{P})$*-flow*. In case such a structure is of the form $\langle \omega, V, \mathit{Succ} \rangle$, where Succ is the standard successor relation on the set ω of natural numbers, we call the structure a *stream model for* P, or a $\wp(\mathsf{P})$*-stream*. $\lhd$

In case the set D of actions is finite, we may just as well identify it with the set $k = \{0, \ldots, k-1\}$, where k is the size of D. We usually restrict to the binary case, that is, $k = 2$. Our main interest will be in Kripke models that are based on the *binary tree*, i.e., a tree in which every node has exactly two successors, a left and a right one.

Definition 1.13 With $2 = \{0, 1\}$, we let 2^* denote the set of finite strings of 0s and 1s. We let ϵ denote the empty string, while the left- and right successor of a node s are denoted by $s \cdot 0$ and $s \cdot 1$, respectively. Written as a relation, we put

$$\mathit{Succ}_i = \{(s, s \cdot i) \mid s \in 2^*\}.$$

A *binary tree over* P, or a *binary* $\wp(\mathsf{P})$*-tree* is a Kripke model of the form $\langle 2^*, V, Succ_0, Succ_1\rangle$. $\lhd$

Remark 1.14 In the general case, the *k-ary tree* is the structure $(k^*, Succ_0, \ldots, Succ_{k-1})$, where k^* is the set of finite sequences of natural numbers smaller than k, and $Succ_i$ is the *i-th successor relation* given by

$$Succ_i = \{(s, s\cdot i) \mid s \in k^*\}.$$

A k-flow model is a Kripke model $\mathbb{S} = \langle S, V, R\rangle$ with k many deterministic accessibility relations, and a *k-ary tree model* is a k-flow model which is based on the k-ary tree. $\lhd$

In deterministic transition systems, the distinction between boxes and diamonds evaporates. It is then convenient to use a single symbol $\bigcirc_i$ to denote either the box or the diamond.

Definition 1.15 The set $\mathrm{MFL}_k(\mathsf{P})$ of formulas of *k-ary Modal Flow Logic* in P is given as follows:

$$\varphi \ ::= \ p \mid \overline{p} \mid \bot \mid \top \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \bigcirc_i \varphi$$

where $p \in \mathsf{P}$, and $i < k$. In case $k = 1$ we will also speak of *modal stream logic*, notation: $\mathrm{MSL}(\mathsf{P})$. $\lhd$

1.2 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 $\mathbb{S}$. This game is an example of a *board game.* In a nutshell, board games are games in which the players move a token along the edge relation of some graph, so that a *match* of *play* of the game corresponds to a (finite or infinite) path through the graph. Furthermore, the winning conditions of a match are determined by the nature of this path. We will meet many examples of board games in these notes, and in Chapter 5 we will study them in more detail.

The evaluation game $\mathcal{E}(\xi, \mathbb{S})$ is played by two *players*: Éloise ($\exists$ or 0) and Abélard ($\forall$ or 1). Given a player Π, we always denote the *opponent* of Π by $\overline{\Pi}$. As mentioned, a *match* of the game consists of the two players moving a *token* from one position to another. *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, in a position of the form $(\Diamond_d \varphi, s)$, it is $\exists$'s turn to move, and she must choose an arbitrary d-successor t of s, thus making (φ, t) the next position. Intuitively, the idea is that in order to show that $\Diamond\varphi$ is true at s, $\exists$ has to come up with a successor of s where φ holds. Formally, we say that the set of *(admissible) next positions* that $\exists$ 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)

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

Table 1: Evaluation game for modal logic

has no admissible next positions, the player whose turn it is to play at (φ, s) *gets stuck* and 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 to 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 $(\overline{p})$ 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.) We may now summarize the game as follows.

Definition 1.16 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, with the set $Sfor(\xi) \times S$ providing the positions of the game; that is, a position is a pair consisting of a subformula of ξ and a point in $\mathbb{S}$. 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 Π in an initialized game is a method that Π uses to select his moves during the play. Such a strategy is *winning for* Π if every match of the game (starting at the given position) is won by Π, provided Π plays according to this strategy. A position (φ, s) is *winning* for Π if Π has a winning strategy for the game initialized in that position. (This is independent of whether it is Π's turn to move at the position.) The set of winning positions in $\mathcal{E}(\xi, \mathbb{S})$ for Π is denoted as $\mathrm{Win}_\Pi(\mathcal{E}(\xi, \mathbb{S}))$.

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

▶ `Example to be added`

Theorem 1.17 (Adequacy) *Let ξ be a modal formula, and let $\mathbb{S}$ be an LTS. Then for any state s in $\mathbb{S}$ it holds that*

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

The proof of this Theorem is left to the reader.

1.3 Bisimulations and bisimilarity

One of the most fundamental notions in the model theory of modal logic is that of a bisimulation between two transition systems.

▶ `discuss bisimilarity as a notion of behavioral equivalence`

Definition 1.18 Let $\mathbb{S}$ and $\mathbb{S}'$ be two transition systems of the same type (P, D). Then a relation $Z \subseteq S \times S'$ is a *bisimulation* of type (P, D) if the following hold, 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: $\mathbb{S}, s \leftrightarroweq_{\mathsf{P},\mathsf{D}} \mathbb{S}', s'$ if there is some bisimulation Z of type (P, D) with $(s, s') \in Z$. If no confusion is likely to arise, we generally drop the subscripts, writing '$\leftrightarroweq$' rather than '$\leftrightarroweq_{\mathsf{P},\mathsf{D}}$'. $\lhd$

Bisimilarity and modal equivalence

In order to understand the importance of this notion for modal logic, the starting point should be the observation that the truth of modal formulas is *invariant* under bisimilarity. Recall that $\equiv$ denotes the relation of modal equivalence.

Theorem 1.19 (Bisimulation Invariance) *Let $\mathbb{S}$ and $\mathbb{S}'$ be two transition systems of the same type. Then*

$$\mathbb{S}, s \leftrightarroweq \mathbb{S}', s' \;\Rightarrow\; \mathbb{S}, s \equiv \mathbb{S}', s'$$

for every pair of states s in $\mathbb{S}$ and s' in $\mathbb{S}'$.

Proof. By a straightforward induction on the complexity of modal formulas one proves that bisimilar states satisfy the same formulas. QED

But there is much more to say about the relation between modal logic and bisimilarity than Theorem 1.19. In particular, for some classes of models, one may prove a converse statement, which amounts to saying that the notions of bisimilarity and modal equivalence coincide. Such classes are said to have the *Hennessy-Milner* property. As an example we mention the class of finitely branching transition systems.

Theorem 1.20 (Hennessy-Milner Property) *Let $\mathbb{S}$ and $\mathbb{S}'$ be two finitely branching transition systems of the same type. Then*

$$\mathbb{S}, s \leftrightarroweq \mathbb{S}', s' \iff \mathbb{S}, s \equiv \mathbb{S}', s'$$

for every pair of states s in $\mathbb{S}$ and s' in $\mathbb{S}'$.

Proof. The direction from left to right follows from Theorem 1.19. In order to prove the opposite direction, one may show that the relation $\equiv$ of modal equivalence itself is a bisimulation. Details are left to the reader. QED

This theorem can be read as indication of the expressiveness of modal logic: any difference in behaviour between two states in finitely branching transition systems can in fact be witnessed by a concrete modal formula. As another witness to this expressivity, in section 1.5 we will see that modal logic is sufficiently rich to express all bisimulation-invariant first-order properties. Obviously, this result also adds considerable strength to the link between modal logic and bisimilarity.

As a corollary of the bisimulation invariance theorem, modal logic has the *tree model property*, that is, every satisfiable modal formula is satisfiable on a structure that has the shape of a tree.

Definition 1.21 A transition system $\mathbb{S}$ of type (P, D) is called *tree-like* if the structure $\langle S, \bigcup_{d \in \mathsf{D}} R_d \rangle$ is a tree. $\lhd$

The key step in the proof of the tree model property of modal logic is the observation that every transition system can be 'unravelled' into a bisimilar tree-like model. The basic idea of such an unravelling is the new states encode (part of) the *history* of the old states. Technically, the new states are the *paths* through the old system.

Definition 1.22 Let $\mathbb{S} = \langle S, V, R \rangle$ be a transition system of type (P, D). A *(finite) path* through $\mathbb{S}$ is a nonempty sequence of the form $(s_0, d_1, s_1, d_2, \ldots, s_n)$ such that $R_{d_i} s_{i-1} s_i$ for all i with $0 < i \leq n$. The set of paths through $\mathbb{S}$ is denoted as $Paths(\mathbb{S})$; we use the notation $Paths_s(\mathbb{S})$ for the set of paths starting at s.

The *unravelling* of $\mathbb{S}$ *around* a state s is the transition system $\vec{\mathbb{S}}_s$ which is coalgebraically defined as the structure $\langle Paths_s(\mathbb{S}), \vec{\sigma} \rangle$, where the coalgebra map $\vec{\sigma} = (\vec{\sigma}_V, (\vec{\sigma}_d \mid d \in \mathsf{D}))$ is given by putting

$$\begin{array}{rcl}\vec{\sigma}_V(s_0, d_1, s_1, d_2, \ldots, s_n) & := & \sigma_V(s_n), \\ \vec{\sigma}_d(s_0, d_1, s_1, d_2, \ldots, s_n) & := & \{(s_0, d_1, s_1, \ldots, s_n, d, t) \in Paths_s(\mathbb{S}) \mid R_d s_n t\}.\end{array}$$

Finally, the unravelling of a pointed transition system $(\mathbb{S}, s)$ is the pointed structure $(\vec{\mathbb{S}}_s, s)$, where (with some abuse of notation) we let s denote the path of length zero that starts and finishes at s. $\lhd$

Clearly, unravellings are tree-like structures, and any pointed transition system is bisimilar to its unravelling. But then the following theorem is immediate by Theorem 1.19.

Theorem 1.23 (Tree Model Property) *Let φ be a satisfiable modal formula. Then φ is satisfiable at the root of a tree-like model.*

Bisimilarity game

We may also give a game-theoretic characterization of the notion of bisimilarity. We first give an informal description of the game that we will employ. 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 the next one. 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* $Z \subseteq S \times S'$ (see below) for s and s'. This relation Z can be seen as providing a set of *witnesses* for $\exists$'s claim that s and s' are bisimilar. Implicitly, $\exists$'s 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 these t and t' are bisimilar.

Definition 1.24 Let $\mathbb{S}$ and $\mathbb{S}'$ be two transition systems of the same type (P, D). Then a relation $Z \subseteq S \times S'$ is a *local bisimulation* for two points $s \in S$ and $s' \in S'$, if it satisfies the properties (prop), (back) and (forth) of Definition 1.18 for this specific s and s':
(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 $(t, t') \in Z$. $\triangleleft$

Note that a local bisimulation for s and s' need only relate successors of s to successors of s'. In particular, the pair (s, s') itself will generally not belong to such a relation. It is easy to see that a relation Z between two Kripke models is a bisimulation iff Z is a local bisimulation for every pair $(s, s') \in Z$.

If a player gets stuck in a match of the bisimilarity game, then the opponent wins the match. For instance, if s and s' disagree about some proposition letter, then there is *no* local bisimulation for s and s', and so the corresponding position (s, s') is an immediate loss for $\exists$. Or, 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, so that $\forall$ 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 1.25 The *bisimilarity game* $\mathcal{B}(\mathbb{S}, \mathbb{S}')$ between two Kripke models $\mathbb{S}$ and $\mathbb{S}'$ is the board game given by Table 2, with the winning condition that finite matches are lost by the player who got stuck, while all infinite matches are won by $\exists$.

A position (s, s') is *winning* for Π if Π has a winning strategy for the game initialized in that position. The set of these positions is denoted as $\mathrm{Win}_\Pi(\mathcal{B}(\mathbb{S}, \mathbb{S}'))$. $\triangleleft$

Also observe that a bisimulation is a relation which is a local bisimulation for each of its members. The following theorem states that the collection of basic winning positions for $\exists$ forms the *largest bisimulation* between $\mathbb{S}$ and $\mathbb{S}'$.

Position	Player	Admissible moves
$(s,s') \in S \times S'$	$\exists$	$\{Z \in \wp(S \times S') \mid Z \text{ is a local bisimulation for } s \text{ and } s'\}$
$Z \in \wp(S \times S')$	$\forall$	$Z = \{(t,t') \mid (t,t') \in Z\}$

Table 2: Bisimilarity game for Kripke models

Theorem 1.26 *Let* $(\mathbb{S}, s)$ *and* $(\mathbb{S}', s')$ *be two pointed Kripke models. Then* $\mathbb{S}, s \leftrightarroweq \mathbb{S}', s'$ *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. A straightforward verification, by induction on the length of the match, shows that this strategy always provides her with a legitimate move, and that it keeps her alive forever. This proves that it is a winning strategy.

For the converse direction, it suffices to show that the relation $\{(t,t') \in S \times S' \mid (t,t') \in \mathrm{Win}_\exists(\mathcal{B}(\mathbb{S},\mathbb{S}'))\}$ itself is in fact a bisimulation. We leave the details for the reader. QED

Remark 1.27 ▶ `The bisimilarity game should not be confused with the` *`bisimulation game`*`.`

◁

Bisimulations via relation lifting

Together, the back- and forth clause of the definition of a bisimulation express that the pair of respective successor sets of two bisimilar states must belong to the so-called *Egli-Milner lifting* $\overline{\wp}Z$ of the bisimulation Z. In fact, the notion of a bisimulation can be completely defined in terms of *relation lifting*.

Definition 1.28 Given a relation $Z \subseteq A \times A'$, define the relation $\overline{\wp}Z \subseteq \wp A \times \wp A'$ as follows:

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

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

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

The relations $\overline{\wp}Z$ and $\overline{\mathsf{K}}Z$ are called the *liftings* of Z with respect to $\wp$ and K, respectively. We say that $Z \subseteq A \times A'$ is *full on* $B \in \wp A$ and $B' \in \wp A'$ if $(B,B') \in \overline{\wp}Z$. ◁

It is completely straightforward to check that a nonempty relation Z linking two transition systems $\mathbb{S}$ and $\mathbb{S}'$ is a local bisimulation for two states s and s' iff $(\sigma(s), \sigma'(s')) \in \overline{\mathsf{K}}Z$. In particular, $\exists$'s move in the bisimilarity game at a position (s,s') consists of choosing a binary relation Z such that $(\sigma(s), \sigma'(s')) \in \overline{\mathsf{K}}Z$. The following characterization of bisimulations is also an immediate consequence.

Proposition 1.29 *Let $\mathbb{S}$ and $\mathbb{S}'$ be two Kripke coalgebras for some Kripke functor K, and let $Z \subseteq S \times S'$ be some relation. Then*

$$Z \text{ is a bisimulation iff } (\sigma(s), \sigma'(s')) \in \overline{\mathsf{K}}Z \text{ for all } (s, s') \in Z. \tag{1}$$

1.4 Finite models and computational aspects

- `complexity of model checking`
- `filtration & polysize model property`
- `complexity of satisfiability`
- `complexity of global consequence`

1.5 Modal logic and first-order logic

- `modal logic is the bisimulation invariant fragment of first-order logic`

1.6 Completeness

1.7 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 a *set* of formulas into a formula.

Definition 1.30 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.* $\lhd$

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

$$\nabla\Phi \equiv \Box \bigvee \Phi \wedge \bigwedge \Diamond\Phi, \tag{2}$$

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):

$$\begin{array}{lcl} \Diamond\varphi & \equiv & \nabla\{\varphi, \top\}, \\ \Box\varphi & \equiv & \nabla\varnothing \vee \nabla\{\varphi\}. \end{array} \tag{3}$$

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

Remark 1.31 Observe that this definition involves the $\forall\exists\&\forall\exists$ 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. Then we find that

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

for any pointed Kripke model $(\mathbb{S}, s)$ and any finite set Φ of formulas. $\lhd$

Given that ∇ and $\{\Diamond, \Box\}$ are mutually expressible, we arrive at the following definition and proposition.

Definition 1.32 Formulas of the language $\mathtt{ML}_\nabla$ are given by the following recursive definition:

$$\varphi \ ::= \ p \mid \overline{p} \mid \bot \mid \top \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \nabla\Phi$$

where Φ denotes a finite set of formulas. $\lhd$

Proposition 1.33 *The languages* $\mathtt{ML}$ *and* $\mathtt{ML}_\nabla$ *are equally expressive.*

Proof. Immediate by (2) and (3). 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 modal distributive law. Recall from Definition 1.28 that a relation $Z \subseteq A \times A'$ is *full on* A and A' if $(A, A') \in \overline{\wp}Z$.

Proposition 1.34 *Let* Φ *and* Φ' *be two sets of formulas. Then the following two formulas are equivalent:*

$$\nabla\Phi \wedge \nabla\Phi' \ \equiv \ \bigvee\{\nabla\Gamma_Z \mid Z \text{ is full on } \Phi \text{ and } \Phi'\}, \tag{4}$$

where, given a relation $Z \subseteq \Phi \times \Phi'$*, we define*

$$\Gamma_Z := \{\varphi \wedge \varphi' \mid (\varphi, \varphi') \in Z\}.$$

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 $\mathbb{S}, s \Vdash \nabla\Gamma_Z$.

The converse direction follows fairly directly from the definitions. QED

As a corollary of Proposition 1.34 we can restrict the use of conjunction in modal logic to that of a *special conjunction* connective $\bullet$ which may only be applied to a propositional formula and a certain conjunction of ∇-formula.

Definition 1.35 Fix finite sets P of proposition letters and D of atomic actions, respectively. We first define the set $\mathtt{CL}(\mathsf{P})$ of *literal conjunctions* by the following grammar:

$$\pi \ ::= \ p \mid \overline{p} \mid \bot \mid \top \mid \pi \wedge \pi.$$

Next, let $\Phi = \{\Phi_d \mid d \in \mathsf{D}\}$ be a D-indexed family of formulas, and write $\nabla_{\mathsf{D}}\Phi := \bigwedge_{d\in\mathsf{D}} \nabla_d \Phi_d$, where ∇_d is the cover modality associated with the accessibility relation R_d of d.

Finally, the following grammar:

$$\varphi \;::=\; \bot \mid \top \mid \varphi \vee \varphi \mid \pi \bullet \nabla_{\mathsf{D}}\Phi.$$

defines the set $\mathrm{DML}_{\mathsf{D}}(\mathsf{P})$ of *disjunctive polymodal formulas* in D and P. $\lhd$

The following theorem states that every modal formula can be rewritten into an equivalent disjunctive normal form.

Theorem 1.36 *For any* P *and* D*, the languages* $\mathtt{ML}_{\mathsf{D}}(\mathsf{P})$ *and* $\mathrm{DML}_{\mathsf{D}}(\mathsf{P})$ *are expressively equivalent.*

We leave the proof of this result as an exercise to the reader.

Notes

Modal logic has a long history in philosophy and mathematics, for an overview we refer to Blackburn, de Rijke and Venema [4]. The use of modal formalisms as specification languages in process theory goes back at least to the 1970s, with Pratt [25] and Pnueli [24] being two influential early papers.

The notion of bisimulation, which plays an important role in modal logic and process theory alike, was first introduced in a modal logic context by van Benthem [3], who proved that modal logic is the bisimulation invariant fragment of first-order logic. The notion was later, but independently, introduced in a process theory setting by Park [23]. At the time of writing we do not know who first took a game-theoretical perspective on the semantics of modal logic. The cover modality ∇ was introduced independently by Moss [19] and Janin & Walukiewicz [12].

Readers who want to study modal logic in more detail are referred to Blackburn, de Rijke and Venema [4] or Chagrov & Zakharyaschev [7].

Exercises

Exercise 1.1 Prove Theorem 1.17.

Exercise 1.2 Prove that the Hennessy-Milner theorem (Theorem 1.20) also holds if only one of the two structures is finitely branching.

Exercise 1.3 (bisimilarity game) Consider the following version $\mathcal{B}_\omega(\mathbb{S}, \mathbb{S}')$ of the bisimilarity game between two transition systems $\mathbb{S}$ and $\mathbb{S}'$. Positions of this game are of the form either $(s, s', \forall, \alpha)$, $(s, s', \exists, \alpha)$ or (Z, α), with $s \in S$, $s' \in S'$, $Z \subseteq S \times S'$ and α either a natural number or ω. The admissible moves for $\exists$ and $\forall$ are displayed in the following table:

Position	Player	Admissible moves
$(s, s', \forall, \alpha)$	$\forall$	$\{(s, s', \exists, \beta) \mid \beta < \alpha\}$
$(s, s', \exists, \alpha)$	$\exists$	$\{(Z, \alpha) \mid Z$ is a local bisimulation for s and $s'\}$
(Z, α)	$\forall$	$\{(s, s', \forall, \alpha) \mid (s, s') \in Z\}$

Note that all matches of this game have finite length.

We write $\mathbb{S}, s \leftrightarroweq_\alpha \mathbb{S}', s'$ to denote that $\exists$ has a winning strategy in the game $\mathcal{B}_\omega(\mathbb{S}, \mathbb{S}')$ starting at position $(s, s', \forall, \alpha)$. It is not hard to see that $\mathbb{S}, s \leftrightarroweq_\omega \mathbb{S}', s'$ iff $\mathbb{S}, s \leftrightarroweq_k \mathbb{S}', s'$ for all $k < \omega$.

(a) Give concrete examples such that $\mathbb{S}, s \leftrightarroweq_\omega \mathbb{S}', s'$ but not $\mathbb{S}, s \leftrightarroweq \mathbb{S}', s'$.
(Hint: think of two modally equivalent but not bisimilar states.)

(b) Let $k \geq 0$ be a natural number. Prove that, for all $\mathbb{S}, s$ and $\mathbb{S}', s'$:

$$\mathbb{S}, s \leftrightarroweq_k \mathbb{S}', s' \;\Rightarrow\; \mathbb{S}, s \equiv_k \mathbb{S}', s'.$$

Here $\equiv_k$ denotes the modal equivalence relation with respect to formulas of modal depth at most k. Here we use a slightly nonstandard notion of modal depth, defined as follows: $d(\bot), d(\top) := 0$, $d(p), d(\overline{p}) := 1$ for $p \in \mathsf{P}$, $d(\varphi \wedge \psi), d(\varphi \vee \psi) := \max(d(\varphi), d(\psi))$, and $d(\Diamond\varphi), d(\Box\varphi) := 1 + d(\varphi)$.

(c) Let $\mathbb{S}$ and $\mathbb{S}'$ be finitely branching transition systems. Prove *directly* (i.e., without using part (b)) that (i) $\Rightarrow$ (ii), for all $s \in S$ and $s' \in S'$:

(i) $\mathbb{S}, s \leftrightarroweq_\omega \mathbb{S}', s'$

(ii) $\mathbb{S}, s \leftrightarroweq \mathbb{S}', s'$.

(d)* Does the implication in (c) hold in the case that only *one* of the two transition systems is finitely branching?

Exercise 1.4 Let Φ and Θ be finite sets of formulas. Prove that

$$\nabla\big(\Phi \cup \{\bigvee \Theta\}\big) \;\equiv\; \bigvee \big\{\nabla\big(\Phi \cup \Theta'\big) \;\mid\; \varnothing \neq \Theta' \subseteq \Theta\big\}.$$

Exercise 1.5 Prove Theorem 1.36.

2 The modal μ-calculus

This chapter is a first introduction to the modal μ-calculus. We define the language, discuss some syntactic issues, and then proceed to its game-theoretic semantics. As a first result, we prove that the modal μ-calculus is bisimulation invariant, and has a strong, 'bounded' version of the tree model property. We then provide some basic information concerning the main complexity measures of μ-calculus formulas: size and alternation depth.

To introduce the formalism, we start with a simple example.

Example 2.1 Consider the formula $\langle d^* \rangle p$ from propositional dynamic logic. By definition, this formula holds at those points in an LTS $\mathbb{S}$ from which there is a finite R_d-path, of unspecified length, leading to a state where p is true.

We leave it for the reader to prove that

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

for any pointed transition system $(\mathbb{S}, s)$ (here we write $\langle d \rangle$ rather than $\Diamond_d$). Informally, one might say that $\langle d^* \rangle p$ is a *fixed point* of the formula $p \vee \langle d \rangle x$, or a solution of the 'equation'

$$x \equiv p \vee \langle d \rangle x. \tag{5}$$

One may show, however, that $\langle d^* \rangle p$ is not the only fixpoint of (5). If we let ∞_d denote a formula that is true at those states of a transition system from which an infinite d-path emanates, then the formula $\langle d^* \rangle p \vee \infty_d$ is another fixed point of (5).

In fact, one may prove that the two mentioned fixpoints are the smallest and largest possible solutions of (5), respectively. $\lhd$

As we will see in this chapter, the modal μ-calculus allows one to explicitly refer to such smallest and largest solutions. For instance, as we will see further on, the smallest and largest solution of the 'equation' (5) will be written as $\mu x.p \vee \langle d \rangle x$ and $\nu x.p \vee \langle d \rangle x$, respectively. Generally, the basic idea underlying the modal μ-calculus is to enrich the language of basic modal logic with two explicit fixpoint operators, μ and ν, respectively. Syntacticlly, these operators behave like quantifiers in first-order logic, in the sense that the application of a fixpoint operator μx to a formula φ *binds* all (free) occurrences of the proposition letter x in φ. The word 'fixpoint' indicates that semantically, the formulas $\mu x\, \varphi$ and $\nu x\, \varphi$ are both 'solutions' to the 'equation' $x \equiv \varphi(x)$, in the sense that, writing $\equiv$ for semantic equivalence, we have both

$$\begin{array}{rrcl} & \mu x\, \varphi & \equiv & \varphi[\mu x\, \varphi / x] \\ \text{and} & \nu x\, \varphi & \equiv & \varphi[\nu x\, \varphi / x], \end{array} \tag{6}$$

where $[\mu x.\varphi / x]$ denotes the operation of substituting $\mu x\, \varphi$ for every free occurrence of x. In other words, both $\mu x\, \varphi$ and $\nu x\, \varphi$ are equivalent to their respective *unfoldings*, $\varphi[\mu x\, \varphi / x]$ and $\varphi[\nu x\, \varphi / x]$.

To arrive at this semantics of modal fixpoint formulas one can take two roads. In Chapter 3 we will introduce the algebraic semantics of $\mu x\, \varphi$ and $\nu x\, \varphi$ in an LTS $\mathbb{S}$, in terms of the *least* and *greatest fixpoint*, respectively, of some algebraically defined meaning function. 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 and a greatest fixpoint. As we will see, this formal definition of the semantics of the modal μ-calculus may be mathematically transparent, but it is of little help when it comes to unravelling and understanding the actual meaning of individual formulas. In practice, it is much easier to work with the *evaluation games* that we will introduce in this chapter.

This framework builds on the game-theoretical semantics for ordinary modal logic as described in Subsection 1.2, extending it with features for the fixpoint operators and for the bound variables of fixpoint formulas (such as x in the formula $\mu x.p \vee \Diamond x$). The key difference lies in the fact that when a match of an evaluation game reaches a position of the form (x, s), with x a *bound* variable, then an equation such as (5) is used to *unfold* the variable x into its associated formula (in the example, the formula $p \vee \Diamond x$).

As a consequence, the flavour of these games is remarkably different from the evaluation games we met before. Recall that in evaluation matches for *basic* modal formulas, the formula is broken down, step by step, until we can declare a winner of the match. From this it follows that the length of such a match is *bounded* by the length of the formula. Evaluation matches for fixpoint formulas, on the other hand, can last forever, if some fixpoint variables are unfolded infinitely often. Hence, the game-theoretic semantics for fixpoint logics takes us to the area of *infinite games*. In this Chapter we keep our treatment of infinite games informal, in Chapter 5 the reader can find precise definitions of all notions that we introduce here.

2.1 Syntax: basics

As announced already in the previous chapter, 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.

Definition 2.2 Given a set D of atomic actions, we define the collection $\mu\mathtt{ML}_{\mathsf{D}}$ of *(poly-)modal fixpoint formulas* as follows:

$$\varphi \ ::= \ \top \mid \bot \mid p \mid \overline{p} \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid \Diamond_d \varphi \mid \Box_d \varphi \mid \mu x\, \varphi \mid \nu x\, \varphi$$

where p and x are propositional variables, and $d \in \mathsf{D}$. There is a restriction on the formation of the formulas $\mu x\, \varphi$ and $\nu x\, \varphi$, namely, that the formula φ is *positive* in x. That is, all occurrences of x in φ are *positive*, or, phrasing it yet differently, no occurrence of x in φ may be in the form of the negative literal $\overline{x}$.

In case the set D of atomic actions is a singleton, we will simply speak of the *modal μ-calculus*, notation: $\mu\mathtt{ML}$.

The syntactic combinations μx and νx are called the *least* and *greatest fixpoint operators*, respectively. We use the symbols η and λ to denote either μ or ν, and we define $\overline{\mu} := \nu$ and $\overline{\nu} := \mu$. A fixpoint formula of the form $\mu x\, \varphi$ is called a *μ-formula*, while *ν-formulas* are the ones of the form $\nu x\, \varphi$. ◁

Convention 2.3 In order to increase readability by reducing the number of brackets, we adopt some standard scope conventions. We let the unary modal connectives, $\Diamond$ and $\Box$, bind stronger than the binary propositional connectives $\wedge$, $\vee$ and $\to$, and use associativity to the left for the connectives $\wedge$ and $\vee$. As an example, we will abbreviate the formula $(\Diamond p \wedge q)$ as $\Diamond p \wedge q$.

Furthermore, we use 'dot notation' to indicate that the fixpoint operators preceding the dot have maximal scope. For instance, $\mu p.\Diamond p \wedge q$ denotes the formula $\mu p\,(\Diamond p \wedge q)$, and not the formula $((\mu p\,\Diamond p) \wedge q)$. As a final example, $\mu x.\overline{p} \vee \Box x \vee y \vee \nu y.q \wedge \Box(x \vee y)$ stands for $\mu x\,\Big(\big((\overline{p} \vee \Box x) \vee y\big) \vee \nu y\,(q \wedge \Box(x \vee y))\Big)$.

Subformulas and free/bound variables

The concepts of *subformula* and *proper subformula* are extended from basic modal logic to the modal μ-calculus in the obvious way.

Definition 2.4 We define the set $\mathit{Sfor}_0(\xi)$ of *direct subformulas* of a formula $\xi \in \mu\mathtt{ML}$ via the following case distinction:

$$\begin{array}{lcll}
\mathit{Sfor}_0(\xi) & := & \varnothing & \text{if } \xi \in \mathtt{At}(\mathsf{P}) \\
\mathit{Sfor}_0(\xi_0 \odot \xi_1) & := & \{\xi_0, \xi_1\} & \text{where } \odot \in \{\wedge, \vee\} \\
\mathit{Sfor}_0(\heartsuit \xi_0) & := & \{\xi_0\} & \text{where } \heartsuit \in \{\Diamond, \Box\} \\
\mathit{Sfor}_0(\eta x.\xi_0) & := & \{\xi_0\} & \text{where } \eta \in \{\mu, \nu\},
\end{array}$$

and we write $\varphi \triangleleft_0 \xi$ if $\varphi \in \mathit{Sfor}_0(\xi)$.

For any formula $\xi \in \mu\mathtt{ML}$, $\mathit{Sfor}(\xi)$ is the least set of formulas which contains ξ and is closed under taking direct subformulas. Elements of the set $\mathit{Sfor}(\xi)$ are called *subformulas* of ξ, and we write $\varphi \trianglelefteq \xi$ ($\varphi \triangleleft \psi$) if φ is a subformula (proper subformula, respectively) of ξ.

The *(subformula) dag* of a formula ξ is defined as the directed acyclic graph $(\mathit{Sfor}(\xi), \triangleright_0)$, where $\triangleright_0$ is the converse of the direct subformula relation $\triangleleft_0$. $\triangleleft$

Syntactically, the fixpoint operators are very similar to the quantifiers of first-order logic in the way they *bind* variables.

Definition 2.5 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:

$$\begin{array}{lcl@{\qquad\qquad}lcl}
FV(\bot) & := & \varnothing & BV(\bot) & := & \varnothing \\
FV(\top) & := & \varnothing & BV(\top) & := & \varnothing \\
FV(p) & := & \{p\} & BV(p) & := & \varnothing \\
FV(\overline{p}) & := & \{p\} & BV(\overline{p}) & := & \varnothing \\
FV(\varphi \vee \psi) & := & FV(\varphi) \cup FV(\psi) & BV(\varphi \vee \psi) & := & BV(\varphi) \cup BV(\psi) \\
FV(\varphi \wedge \psi) & := & FV(\varphi) \cup FV(\psi) & BV(\varphi \wedge \psi) & := & BV(\varphi) \cup BV(\psi) \\
FV(\Diamond_d \varphi) & := & FV(\varphi) & BV(\Diamond_d \varphi) & := & BV(\varphi) \\
FV(\Box_d \varphi) & := & FV(\varphi) & BV(\Box_d \varphi) & := & BV(\varphi) \\
FV(\eta x.\varphi) & := & FV(\varphi) \setminus \{x\} & BV(\eta x.\varphi) & := & BV(\varphi) \cup \{x\}
\end{array}$$

For a finite set of propositional variables P, we let $\mu\mathtt{ML}_{\mathsf{D}}(\mathsf{P})$ denote the set of $\mu\mathtt{ML}_{\mathsf{D}}$-formulas φ of which all free variables belong to P. $\triangleleft$

Formulas like $x \vee \mu x.((p \vee x) \wedge \Box \nu x.\Diamond x)$ may be well formed, but in practice they are very hard to read and to work with. In the sequel we will often work with formulas in which every bound variable uniquely determines a subformula where it is bound, and almost exclusively with formulas in which no variable has both free and bound occurrences in φ.

Definition 2.6 A formula $\varphi \in \mu\mathtt{ML}_{\mathsf{D}}$ is *tidy* if $FV(\varphi) \cap BV(\varphi) = \varnothing$, and *clean* if in addition with every bound variable x of φ we may associate a unique subformula of the form $\eta x.\delta$. In the latter case we let $\varphi_x = \eta_x x.\delta_x$ denote this unique subformula. $\lhd$

Convention 2.7 As a notational convention, we will use the letters $p, q, r, \ldots$ and $x, y, z, \ldots$ to denote, respectively, the free and the bound propositional variables of a $\mu\mathtt{ML}_{\mathsf{D}}$-formula. This convention can be no more than a guideline, since the division between bound and free variables may not be the same for a formula and its subformulas. For instance, the variable x is bound in $\mu x.p \vee \Diamond x$, but free in its subformula $p \vee \Diamond x$.

Substitution & unfolding

The syntactic operation of substitution is ubiquitous in any account of the modal μ-calculus, first of all because it features in the basic operation of unfolding a fixpoint formula. As usual in the context of a formal language featuring operators that *bind* variables, the definition of a substitution operation needs some care.

In particular, we want to protect the substitution operation from variable capture. To give a concrete example, suppose that we would naively define a substitution operation ψ/x by defining $\varphi[\psi/x]$ to be the formula we obtain from the formula φ by replacing every free occurrences of x with the formula ψ. Now consider the formula $\varphi(q) = \mu p.q \vee \Diamond p$ expressing the reachability of a q-state in finitely many steps. If we substitute p for q in φ, we would expect the resulting formula to express the reachability of a p-state in finitely many steps, but the formula we obtain is $\varphi[p/q] = \mu p.p \vee \Diamond p$, which says something rather different (in fact, it happens to be equivalent to $\bot$). Even worse, the substitution $[\overline{p}/q]$ would produce a syntactic string $\varphi[\overline{p}/q] = \mu p.\overline{p} \vee \Diamond p$ which is not even a well-formed formula.

To avoid such anomalies, for the time being we shall only consider substitutions ψ/x applied to formulas where ψ is free for x.

Definition 2.8 Let ψ, ξ and x be respectively two modal μ-calculus formulas and a propositional variable. We say that ψ is *free for x in ξ* if ξ is positive in x[1] and for every variable $y \in FV(\psi)$, every occurrence of x in a subformula $\eta y.\chi$ of ξ is in the scope of a fixpoint operator λx in ξ, i.e., bound in ξ by some occurrence of λx. $\lhd$

Definition 2.9 Let $\{\psi_z \mid z \in Z\}$ be a set of modal μ-calculus formulas, indexed by a set of variables Z, let $\varphi \in \mu\mathtt{ML}$ be positive in each $z \in Z$, and assume that each ψ_z is free for z in φ.

[1] Strictly speaking, this condition is not needed. In particular, as a separate atomic case of our inductive definition, we could define the outcome of the substitution $\overline{p}[\psi/p]$ to be the *negation* of the formula ψ (suitably defined). However, we will only need to look at substitutions $\varphi[\psi/z]$ where we happen to know that φ is positive in z. As a result, our simplified definition does not impose a real restriction.

We inductively define the *simultaneous substitution* $[\psi_z/z \mid z \in Z]$ as the following operation on $\mu\mathtt{ML}$:

$$\begin{array}{lcl}
\varphi[\psi_z/z \mid z \in Z] & := & \begin{cases} \psi_p & \text{if } \varphi = p \in Z \\ \varphi & \text{if } \varphi \text{ is atomic but } \varphi \notin Z \end{cases} \\
(\heartsuit\varphi)[\psi_z/z \mid z \in Z] & := & \heartsuit\varphi[\psi_z/z \mid z \in Z] \\
(\varphi_0 \odot \varphi_1)[\psi_z/z \mid z \in Z] & := & \varphi_0[\psi_z/z \mid z \in Z] \odot \varphi_1[\psi_z/z \mid z \in Z] \\
(\eta x.\varphi)[\psi_z/z \mid z \in Z] & := & \eta x.\varphi[\psi_z/z \mid z \in Z \setminus \{x\}]
\end{array}$$

In case Z is a singleton, say $Z = \{z\}$, we will simply write $\varphi[\psi_z/z]$. $\lhd$

Remark 2.10 In case ψ is not free for some $z \in Z$ in ξ, we can define a correct version of the substitution $\xi[\psi/x]$ by taking some (canonically chosen) *alphabetical variant* ξ' of ξ such that each ψ_z is free for z in ξ', and setting

$$\xi[\psi_z/z \mid z \in Z] := \xi'[\psi_z/z \mid z \in Z].$$

Note however, that the operation of taking alphabetical variants requires some attention, since it comes at a price in terms of the *size* of the formula. We will come back to this matter in more detail later. $\lhd$

The reason that the modal μ-calculus, and related formalisms, are called *fixpoint logics* is that, for $\eta = \mu/\nu$, the meaning of the formula $\eta x.\chi$ in a model $\mathbb{S}$ is given as the least/greatest *fixpoint* of the semantic map $\chi_x^{\mathbb{S}}$ expressing the dependence of the meaning of χ on (the meaning of) the variable x. As a consequence, the following equivalence lies at the heart of semantics of $\mu\mathtt{ML}$:

$$\eta x.\chi \equiv \chi[\eta x.\chi/x] \tag{7}$$

Definition 2.11 Given a formula $\eta x.\chi \in \mu\mathtt{ML}$, we call the formula $\chi[\eta x.\chi/x]$ its *unfolding*. $\lhd$

Remark 2.12 Unfolding is the central operation in taking the closure of a formula that we are about to define. Unfortunately, the collection of clean formulas is not closed under unfolding. Consider for instance the formula $\varphi(p) = \nu q.\Diamond q \wedge p$, then we see that the formula $\mu p.\varphi$ is clean, but its unfolding $\varphi[\mu p.\varphi/p] = \nu q.\Diamond q \wedge \mu p\, \nu q.\Diamond q \wedge p$ is not. Furthermore, our earlier observation that the naive version of substitution may produce 'formulas' that are not well-formed applies here as well. For instance, with χ denoting the formula $\overline{p} \wedge \nu p.\Box(x \vee p)$, unfolding the formula $\mu x.\chi$ would produce the ungrammatical $\overline{p}\wedge\nu p.\Box\big((\mu x.\overline{p}\wedge\nu p.\Box(x\vee p))\vee p\big)$. $\lhd$

Fortunately, the condition of *tidyness* guarantees that we may calculate unfoldings without moving to alphabetical variants, since we can prove that the formula $\eta x.\chi$ is free for x in χ, whenever $\eta x.\chi$ is tidy. In addition, tidyness is preserved under taking unfoldings.

Proposition 2.13 *Let $\eta x.\chi \in \mu\mathtt{ML}$ be a tidy formula. Then*

1) $\eta x.\chi$ is free for x in χ;

2) $\chi[\eta x.\chi/x]$ is tidy as well.

Proof. For part 1), take a variable $y \in FV(\eta x.\chi)$. Then obviously y is distinct from x, while $y \notin BV(\eta x.\chi)$ by tidyness. Clearly then we find $y \notin BV(\chi)$; in other words, χ has *no* subformula of the form $\lambda y.\psi$. Hence it trivially follows that $\eta x.\chi$ is free for x in χ.

Part 2) is immediate by the following identities:

$$\begin{array}{lllll} FV(\chi[\eta x.\chi/x]) & = & (FV(\chi) \setminus \{x\}) \cup FV(\eta x.\chi) & = & FV(\eta x.\chi) \\ BV(\chi[\eta x.\chi/x]) & = & BV(\chi) \cup BV(\eta x.\chi) & = & BV(\eta x.\chi) \end{array}$$

which can easily be proved. QED

Dependency order & guardedness

An important role in the theory of the modal μ-calculus is played by a certain order $\leq_\xi$ on the bound variables of a formula ξ, with $x \leq y$ indicating that y is 'more significant' than x, in the sense that the meaning of x/δ_x is (in principle) dependent on the meaning of y/δ_y. The key situation where this happens is when y occurs freely in δ_x. Observe that this can only be the case if $\delta_x \trianglelefteq \delta_y$, so that the relation '$y$ occurs freely in δ_x' does not have any cycles, and thus naturally induces a partial order.

Definition 2.14 Given a clean formula ξ, we define a *dependency order* $\leq_\xi$ on the set $BV(\xi)$, saying that y *ranks higher* than x if $x \leq_\xi y$. The relation $\leq_\xi$ is defined as the least partial order containing all pairs (x, y) such that $y \trianglelefteq \delta_x \trianglelefteq \delta_y$. $\triangleleft$

We finish our sequence of basic syntactic definitions with the notion of guardedness, which will become important later on.

Definition 2.15 A variable x is *guarded* in a $\mu\mathtt{ML}_\mathsf{D}$-formula φ if every occurrence of x in φ is in the scope of a modal operator. A formula $\xi \in \mu\mathtt{ML}_\mathsf{D}$ is *guarded* if for every subformula of ξ of the form $\eta x.\delta$, x is guarded in δ. $\triangleleft$

In the next chapter we will prove that every formula can be effectively rewritten into an equivalent, clean and guarded formula.

2.2 Game semantics

For a definition of the evaluation game of the modal μ-calculus, fix a *clean* formula ξ and an LTS $\mathbb{S}$. 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 2.16 Given a clean modal μ-calculus formula ξ and a transition system $\mathbb{S}$, we define the *evaluation game* or *model checking game* $\mathcal{E}(\xi, \mathbb{S})$ as 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.

As before, $\mathcal{E}(\xi, \mathbb{S})@(\xi, s)$ denotes the instantiation of this game where the starting position is fixed as (ξ, s). $\triangleleft$

One might expect that the main difference with the evaluation game for basic modal formulas would involve the new formula constructors of the μ-calculus: the fixpoint operators. Perhaps surprisingly, we can deal with the fixpoint operators themselves in the most straightforward way possible, viz., by simply *stripping* them. That is, the successor of a position of the form $(\eta x.\delta, s)$ is simply obtained as the pair (δ, s). Since this next position is thus uniquely determined, the position $(\eta x.\delta, s)$ will not be assigned to either of the players.

The crucial difference lies in the treatment of the *bound variables* of a fixpoint formula ξ. Previously, all positions of the form (p, s) would be *final positions* of the game, immediately determining the winner of the match, and this is still the case here if p is a *free* variable. 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 variables 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 assigned to neither of the players.

Position	Player	Admissible moves
$(\varphi_1 \vee \varphi_2, s)$	$\exists$	$\{(\varphi_1, s), (\varphi_2, s)\}$
$(\varphi_1 \wedge \varphi_2, s)$	$\forall$	$\{(\varphi_1, s), (\varphi_2, s)\}$
$(\Diamond_d \varphi, s)$	$\exists$	$\{(\varphi, t) \mid t \in \sigma_d(s)\}$
$(\Box_d \varphi, s)$	$\forall$	$\{(\varphi, t) \mid t \in \sigma_d(s)\}$
$(\bot, s)$	$\exists$	$\varnothing$
$(\top, s)$	$\forall$	$\varnothing$
(p, s), with $p \in FV(\xi)$ and $s \in V(p)$	$\forall$	$\varnothing$
(p, s), with $p \in FV(\xi)$ and $s \notin V(p)$	$\exists$	$\varnothing$
$(\overline{p}, s)$, with $p \in FV(\xi)$ and $s \in V(p)$	$\exists$	$\varnothing$
$(\overline{p}, s)$, 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 2.17 Let $\mathbb{S} = \langle S, R, V\rangle$ be the Kripke model based on the set $S = \{0, 1, 2\}$, with $R = \{(0, 1), (1, 1), (1, 2), (2, 2)\}$, and V 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 $\exists$. Assuming that she wants to win, 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. Choosing $(x, 2)$, however, 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 already visited by the match 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. Here we arrive at one of the key ideas underlying the semantics of fixpoint formulas, which in a slogan can be formulated as follows:

$\boxed{\nu \text{ means unfolding, } \mu \text{ means finite unfolding.}}$

Giving a more detailed interpretation to this slogan, in case of a unique variable that is unfolded infinitely often during a match Σ, we will declare $\exists$ to be the winner of Σ if this variable is a ν-variable, and $\forall$ in case we are dealing with a μ-variable. But what happens in case that *various* variables are unfolded infinitely often? As we shall see, in these cases there is always a *unique* such variable that ranks higher than any other such variable.

Definition 2.18 Let ξ be a clean $\mu\mathtt{ML}_{\mathsf{D}}$-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

$$\Sigma = (\varphi_i, s_i)_{i<\kappa}$$

(where κ is either a natural number or ω) which is in accordance with the rules of the evaluation game — that is, Σ is a path through the game graph given by the admissibility relation 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 Σ, we let $Unf^{\infty}(\Sigma) \subseteq BV(\xi)$ denote the set of variables that are unfolded infinitely often during Σ. $\lhd$

Proposition 2.19 *Let ξ be a clean $\mu\mathtt{ML}_{\mathsf{D}}$-formula, and $\mathbb{S}$ a labelled transition system. Then for any infinite match Σ of the game $\mathcal{E}(\xi, \mathbb{S})$, the set $Unf^{\infty}(\Sigma)$ has a highest ranking member, in terms of the dependency order of Definition 2.14.*

Proof. Since Σ is an infinite match, the set $U := \mathit{Unf}^\infty(\Sigma)$ is not empty. Let y be an element of U which is *maximal* (with respect to the ranking order $\leq_\xi$) — such an element exists since U is finite. We claim that

$$\text{from some moment on, } \Sigma \text{ only features subformulas of } \delta_y. \tag{8}$$

To prove this, note that since y is $\leq_\xi$-maximal in U, there must be a position in Σ such that y is unfolded to δ_y, while no variable $z >_\xi y$ is unfolded at any later position in Σ. But then a straightforward induction shows that all formulas featuring at later positions must be subformulas of δ_y: the key observation here is that if $z \trianglelefteq \delta_y$ unfolds to δ_z, and by assumption $z \not>_\xi y$, then it must be the case that $\delta_z \trianglelefteq \delta_y$.

As a corollary of (8), we claim that

$$y \text{ is in fact the maximum of } U \text{ (with respect to } \leq_\xi). \tag{9}$$

To see this, suppose for contradiction that there is a variable $x \in U$ which is *not* below y. It follows from (8) that $\delta_x \trianglelefteq \delta_y$, and without loss of generality we may assume x to be such that δ_x is a *maximal* subformula of δ_y such that $x \not\leq_\xi y$ (in the sense that $z \leq_\xi y$ for all $z \in U$ with $\delta_x \triangleleft \delta_z$.) In particular then we have $y \notin FV(\delta_x)$. But since y is unfolded infinitely often, there must be a variable $z \in FV(\delta_x)$ which allows Σ to 'leave' δ_x infinitely often; this means that $z \in U$, $\delta_x \trianglelefteq \delta_z$ but $\delta_z \ntrianglelefteq \delta_x$. From this it is immediate that $x \leq_\xi z$, while from $z \in U$ and (8) we obtain $\delta_z \trianglelefteq \delta_y$. It now follows from our maximality assumption on x that $z \leq_\xi y$. But then by transitivity of $\leq_\xi$ we find that $x \leq_\xi y$ indeed. In other words, we have arrived at the desired contradiction.

This shows that (9) holds indeed, and from this the Proposition is immediate. QED

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

Definition 2.20 Let ξ be a clean $\mu\mathtt{ML}_\mathsf{D}$-formula. The winning conditions of the game $\mathcal{E}(\xi, \mathbb{S})$ are given in Table 4. $\triangleleft$

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

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

We can now formulate the game-theoretic semantics of the modal μ-calculus as follows.

Definition 2.21 Let ξ be a clean formula of the modal μ-calculus, and let $\mathbb{S}$ be a transition system of the appropriate type. Then we say that ξ is (game-theoretically) *satisfied* at s, notation: $\mathbb{S}, s \Vdash_g \xi$ if $(\xi, s) \in \mathrm{Win}_\exists(\mathcal{E}(\xi, \mathbb{S}))$. $\triangleleft$

Remark 2.22 As mentioned we have kept this introduction to evaluation games for fixpoint formulas rather informal, referring to Chapter 5 for a more rigorous discussion of infinite games. Nevertheless, we want to mention already here that evaluation games, on the ground of being so-called *parity games*, have two very useful properties that make them attractive to work with. To start with, every evaluation game is *determined* in the sense that every position is winning for exactly one of the two players. And second, one may show that winning strategies for either player of an evaluation game, can always be assumed to be *positional*, that is, do not depend on moves made earlier in the match, but only on the current position. Combining this, evaluation games enjoy *positional determinacy*; that is, every position (φ, s) is winning for exactly one of the two players, and each player $\Pi \in \{\exists, \forall\}$ has a *positional* strategy f_Π which is winning for the game $\mathcal{E}(\xi, \mathbb{S})@(\varphi, s)$ for every position (φ, s) that is winning for Π. $\lhd$

Remark 2.23 Observe that we have defined the game-theoretic semantics for *clean* formula only. In the next section we define an alternative version of the evaluation game which works for arbitrary *tidy* formulas.

It is certainly possible to extend this definition to arbitrary fixpoint formulas; a straightforward approach would be to involve the *construction tree* of a non-clean formula ξ, and redefine a *position* of the evaluation game $\mathcal{E}(\xi, \mathbb{S})$ to be a pair, consisting of a node in this construction tree and a point in the Kripke structure. Alternatively, one may work with a clean *alphabetical variant* of the formula ξ; once we have given the algebraic semantics for arbitrary formulas, it is not hard to show that in that semantics, alphabetic variants are equivalent. $\lhd$

2.3 Examples

Example 2.24 As a first example, consider the formulas $\eta x.p \vee x$, and fix a Kripke model $\mathbb{S}$. Observe that any match of the evaluation game $\mathcal{E}(\eta x.p \vee x, \mathbb{S})$ starting at position $(\eta x.p \vee x, s)$ immediately proceeds to position $(p \vee x, s)$, after which $\exists$ can make a choice. In case η is the least fixpoint operator, $\eta = \mu$, we claim that

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

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 $s \in \mathrm{Win}_\exists(\mathcal{E}(\mu x.p \vee x, \mathbb{S}))$.

On the other hand, if $s \notin V(p)$, then $\exists$ will lose if she chooses the 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. Hence in this case we see that $s \notin \mathrm{Win}_\exists(\mathcal{E}(\nu x.p \vee x, \mathbb{S}))$.

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

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

It is easy to see that now, 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 $\exists$.

Concluding, we see that $\mu x.p \vee x$ is equivalent to the formula p, and $\nu x.p \vee x$, to the formula $\top$. $\lhd$

Example 2.25 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 $s_0Rs_1Rs_2R\dots$ through $\mathbb{S}$.
2. the match is a 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 $s_0Rs_1R\dots s_k$ 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
- $\mu x.\Diamond x$ is equivalent to the formula $\bot$,
- $\mathbb{S}, s \Vdash_g \nu x.\Diamond x$ iff there is an infinite path starting at s. $\lhd$

► `Until operator`

The examples that we have considered so far involved only a single fixpoint operator. We now look at an example containing both a least and a greatest fixpoint operator.

Example 2.26 Let ξ be the following formula:
$$\xi = \nu x.\mu y.\underbrace{(p \wedge \Diamond x)}_{\alpha_p} \vee \underbrace{(\overline{p} \wedge \Diamond y)}_{\alpha_{\overline{p}}}$$

Then we claim that for any LTS $\mathbb{S}$, and any state s in $\mathbb{S}$:
$$\mathbb{S}, s \Vdash_g \xi \text{ iff there is some path from } s \text{ on which } p \text{ is true infinitely often.} \tag{10}$$

To see this, first suppose that there is a path $\Sigma = s_0s_1s_2\dots$ as described in the right hand side of (10) and suppose that $\exists$ plays according to the following strategy:

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

(b) at a position $(\Diamond\varphi, t)$, distinguish cases:
- if the match so far has followed the path, with $t = s_k$, choose (φ, s_{k+1});
- otherwise, choose an arbitrary successor (if possible).

We claim that this is a winning strategy for $\exists$ in the evaluation game initialized at (ξ, s). Indeed, since $\exists$ always chooses the propositionally safe disjunct of $\alpha_p \vee \alpha_{\overline{p}}$, 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 to always get to positions of the form $(\Diamond z, s_i)$, and thus she can force the match to last infinitely long, following the infinite path Σ. But why does she actually *win* this match? The point is that, whenever she chooses α_p, three positions later, x will be unfolded, and likewise with $\alpha_{\overline{p}}$ and y. Thus, p being true infinitely often on Σ means that the ν-variable x gets unfolded infinitely often. And so, even though the μ-variable y might get unfolded infinitely often as well, she wins the match since x ranks higher than y anyway.

For the other direction, assume that $\mathbb{S}, s \Vdash_g \xi$ 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 $(\overline{p} \wedge \Diamond z, t)$. 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 both players will stay alive forever: $\forall$ because he can always choose a diamond conjunct, and $\exists$ because we assumed her strategy to be winning. But a second consequence of $\exists$ playing a winning strategy, 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. $\lhd$

2.4 Bounded 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 μ–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 2.27 (Bisimulation Invariance) *Let ξ be a modal fixpoint formula with $FV(\xi) \subseteq$ P, and let $\mathbb{S}$ and $\mathbb{S}'$ be two labelled transition systems with points s and s', respectively. If $\mathbb{S}, s \leftrightarroweq_{\mathsf{P}} \mathbb{S}', s'$, then*

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

Proof. Assume that $s \leftrightarroweq_{\mathsf{P}} s'$ and that $\mathbb{S}, s \Vdash_g \xi$, with $FV(\xi) \subseteq \mathsf{P}$. We will show that $\mathbb{S}', s' \Vdash_g \xi$. By definition we may assume that $\exists$ has a winning strategy f in the evaluation game $\mathcal{E} := \mathcal{E}(\xi, \mathbb{S})$ initialized at (ξ, s); that is, given an f-guided partial $\mathcal{E}$-match Σ ending in a position for $\exists$, we let $f(\Sigma)$ denote the next position as determined by f.

We need 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 $s \leftrightarroweq s'$ instead of $s \leftrightarroweq_{\mathsf{P}} s'$).

- In case ξ is a literal, 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 a literal, 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 $s \leftrightarroweq s'$, there is a successor t' of s' such that $t \leftrightarroweq t'$. 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 loses, 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 $t \leftrightarroweq t'$. 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 $u \leftrightarroweq u'$, and such that the move in $\mathcal{E}$ is consistent with f. We may in fact show that she can maintain this condition throughout the match, and it is not hard to see that she can construct a winning strategy based on this.

Making this proof sketch a bit more precise, we introduce some terminology (anticipating the formal treatment of games in Chapter 5). Generally we identify *matches* of a game with certain sequences of positions in that game, and we say that a match $\Sigma = p_0 p_1 \dots p_n$ is *guided* by a strategy f for player $\Pi \in \{\exists, \forall\}$ if for every $i < n$ such that position p_i belongs to Π, the next position p_{i+1} is indeed the position dictated by the strategy f. In the context of this particular proof we say that an $\mathcal{E}'$-match $\Sigma' = (\varphi'_0, s'_0)(\varphi'_1, s'_1) \dots (\varphi'_n, s'_n)$ is *linked* to an $\mathcal{E}$-match $\Sigma = (\varphi_0, s_0)(\varphi_1, s_1) \dots (\varphi_n, s_n)$ (of the same length), if $\varphi'_i = \varphi_i$ and $\mathbb{S}', s'_i \leftrightarroweq \mathbb{S}, s_i$ for all i with $0 \leq i \leq n$. The key claim in the proof states that, for a $\mathcal{E}'$-match Σ', if $\exists$ has established such a bisimilarity link with an $\mathcal{E}$-match that is f-guided, then she will either win the $\mathcal{E}'$-game immediately, or else she can maintain the link during one further round of the game.

Claim 1 Let Σ' be a finite $\mathcal{E}'$-match, and assume that Σ is linked to some f-guided $\mathcal{E}$-match Σ. Then one of the following two cases apply.

1) both $last(\Sigma')$ and $last(\Sigma)$ are positions for $\exists$, and $\exists$ can continue Σ' with a legitimate move (φ, t') such that $\Sigma' \cdot (\varphi, t')$ is bisimilarity-linked to $\Sigma \cdot (\varphi, t)$, where (φ, t) is the move dicated by f in Σ.

2) both $last(\Sigma')$ and $last(\Sigma)$ are positions for $\forall$, and for every move (t, φ') for $\forall$ in Σ' there is a legitimate move (φ, t) for $\forall$ in Σ such that $\Sigma' \cdot (\varphi, t')$ is bisimilarity-linked to $\Sigma \cdot (\varphi, t)$.

The *proof* of this Claim proceeds via an obvious adaptation of the case-by-case argument just given for the initial positions of $\mathcal{E}'$ and $\mathcal{E}$. Omitting the details, we move on to show that based on Claim 1, $\exists$ has a winning strategy in $\mathcal{E}'$.

By a straightforward inductive argument we may provide $\exists$ with a strategy f' in $\mathcal{E}'$, and show how to maintain, simultaneously, for every f'-guided match Σ, an f-guided $\mathcal{E}$-match which is linked to Σ'. For the base case of this induction, simply observe that by the assumption that $\mathbb{S}, s \leftrightarroweq \mathbb{S}', s'$, the initial positions of $\mathcal{E}'$ and $\mathcal{E}$ constitute linked (trivial) matches. For the inductive case we consider an f'-guided $\mathcal{E}'$-match Σ', and inductively assume that there is a bisimilarity-linked f-guided $\mathcal{E}$-match Σ. Now distinguish cases:

- If $last(\Sigma')$ is a position for $\exists$, we use item 1) of Claim 1 to define her move (φ, t'); it follows that $\Sigma' \cdot (\varphi, t')$ and $\Sigma \cdot (\varphi, t)$ are bisimilarity-linked (where (φ, t) is the move dicated by f in Σ).

- On the other hand, in case $last(\Sigma')$ is a position for $\forall$, assume that he makes some move, say, (t', ψ); now we use item 2) of the claim to define a continuation $\Sigma \cdot (t, \psi)$ of Σ that is bisimilarity-linked to $\Sigma' \cdot (t', \psi)$.

To see why the strategy f' of $\exists$ is *winning* for her, consider a *full* (i.e., finished) f'-guided match Σ', and distinguish cases. If Σ' is finite, this means that one of the players must be stuck, and we have to show that this player must be $\forall$. But we just showed that there must be an f-guided match Σ which is bisimilarity-linked to Σ'. It follows from the definiton of linked matches that the final positions of Σ' and Σ must be, respectively, of the form (φ', t) and (φ, t) for some formula φ and states t', t such that $\mathbb{S}', t' \leftrightarroweq \mathbb{S}, t$. From this it is not hard to derive that the same player who got stuck in Σ' also got stuck in Σ; and since Σ is guided by $\exists$'s supposedly winning strategy f, this player must be $\forall$ indeed.

If Σ is infinite, say $\Sigma' = (\varphi_i, s'_i)_{i<\omega}$, the shadow $\mathcal{E}$-match maintained by $\exists$ is infinite as well. More precisely, the inductive argument given above reveals the existence of an infinite, f-guided $\mathcal{E}$-match $\Sigma = (\varphi_i, s_i)_{i<\omega}$ such that $\mathbb{S}', s'_i \leftrightarroweq \mathbb{S}, s_i$ for all $i < \omega$. The key observation, however, is that the two sequences of formulas, in the $\mathcal{E}'$-match Σ' and its $\mathcal{E}$-shadow Σ, respectively, are exactly the *same*. This means that also in the infinite case the winner of Σ' is the winner of Σ, and since Σ is f-guided, this winner must be $\exists$. QED

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

Theorem 2.28 (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 $\mathbb{S}$. Then by bisimulation invariance, ξ is satisfiable at the root of the *unravelling* $\vec{\mathbb{S}}_s$ of $\mathbb{S}$ around s, cf. Definition 1.22. This unravelling clearly is a tree model. QED

For the next theorem, recall that the size of a formula is simply defined as the number of its subformulas.

Theorem 2.29 (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 $\exists$ has a winning strategy f in the game $\mathcal{E}@(\xi, r)$, where we abbreviate $\mathcal{E} := \mathcal{E}(\xi, \mathbb{T})$. By the Positional Determinacy of the evaluation game, we may assume that this strategy is positional — this will simplify our argument a bit. We may thus represent this strategy as a map f that, among other things, maps positions of the form $(\Diamond\varphi, s)$ to positions of the form (φ, t) with Rst.

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. (Note that $\mathbb{T}_\omega$ is in general not a generated submodel of $\mathbb{T}$: not all successors of nodes in $\mathbb{T}_\omega$ need to belong to $\mathbb{T}_\omega$.) From the construction it is obvious that the branching degree of $\mathbb{T}_\omega$ is bounded by the size of ξ, because ξ has at most $|\xi|$ diamond subformulas.

We claim that $\mathbb{T}_\omega, r \Vdash_g \xi$. 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 $\Sigma = (\xi, r)(\varphi_1, t_1)\dots$ which is consistent with f'. The key observation of the proof is that Σ 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 Σ 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 Σ in $\mathcal{E}$. But then Σ 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, Σ is also winning as an $\mathcal{E}'$-match. And since Σ was an arbitrary $\mathcal{E}'$-match starting at (ξ, r), this shows that f' is a winning strategy, as required. QED

2.5 Size

Concerning the complexity of a modal μ-calculus formula, we will see that two measures feature prominently: its size and its alternation depth. Both notions are in fact quite subtle in that they admit several non-equivalent definitions.

Length and dag-size

For the definition of the *size* of a formula at first sight there seem to be two natural candidates.

Definition 2.30 Given a μ-calculus formula ξ, we define its *length* $|\xi|^\ell$ as the number of (non-negation) symbols occurring in ξ:

$$\begin{array}{lcll}
|\varphi|^\ell & := & 1 & \text{if } \varphi \text{ is atomic} \\
|\varphi_0 \odot \varphi_1|^\ell & := & 1 + |\varphi_0|^\ell + |\varphi_1|^\ell & \text{where } \odot \in \{\wedge, \vee\} \\
|\heartsuit\varphi|^\ell & := & 1 + |\varphi|^\ell & \text{where } \heartsuit \in \{\Diamond, \Box\} \\
|\eta x.\varphi|^\ell & := & 2 + |\varphi|^\ell & \text{where } \eta \in \{\mu, \nu\}
\end{array}$$

The *dag-size* of ξ is defined as follows:

$$|\xi|^d := |Sfor(\xi)|,$$

i.e., $|\xi|^d$ is given as the number of subformulas of ξ. $\lhd$

Clearly, the terminology 'dag-size' is explained by the fact that the dag-size of a formula corresponds to the number of vertices in its subformula dag.

▶ `compare length and dag-size`

Closure

However, as we will motivate further on, it is more natural to define the size of a μ-calculus formula in terms of its *closure set* rather than of its collection of subformulas. In words, $Clos(\xi)$ is the smallest set which contains ξ and is closed under direct boolean and modal subformulas, and under unfoldings of fixpoint formulas. It will be convenient to define this set in terms of so-called *traces*.

Definition 2.31 Let $\to_C$ be the binary relation between tidy μ-calculus formulas given by the following exhaustive list:

1) $(\varphi_0 \odot \varphi_1) \to_C \varphi_i$, for any $\varphi_0, \varphi_1 \in \mu\mathtt{ML}$, $\odot \in \{\wedge, \vee\}$ and $i \in \{0, 1\}$;
2) $\heartsuit\varphi \to_C \varphi$, for any $\varphi \in \mu\mathtt{ML}$ and $\heartsuit \in \{\Diamond, \Box\}$);
3) $\eta x.\varphi \to_C \varphi[\eta x.\varphi/x]$, for any $\eta x.\varphi \in \mu\mathtt{ML}$, with $\eta \in \{\mu, \nu\}$.

We call a $\to_C$-path $\psi_0 \to_C \psi_1 \to_C \cdots \to_C \psi_n$ a *(finite) trace*; similarly, an *infinite trace* is a sequence $(\psi_i)_{i<\omega}$ such that $\psi_i \to_C \psi_{i+1}$ for all $i < \omega$.

We define the relation $\twoheadrightarrow_C$ as the reflexive and transitive closure of $\to_C$, and define the *closure* of a formula ψ as the set

$$Clos(\psi) := \{\varphi \mid \psi \twoheadrightarrow_C \varphi\}.$$

Given a set of formulas Ψ, we put $Clos(\Psi) := \bigcup_{\psi \in \Psi} Clos(\psi)$. Formulas in the set $Clos(\psi)$ are said to be *derived* from ψ. The *closure graph* of ψ is the directed graph $(Clos(\xi), \to_C)$. $\lhd$

Clearly then, a formula χ belongs to the closure of a formula ψ iff there is a trace from ψ to χ. This trace perspective will be particularly useful when we need to prove statements about the formulas belonging to the closure of a certain formula. We will occasionally think of Definition 2.31 as a derivation system for statements of the form $\varphi \in Clos(\psi)$, and of a trace $\psi = \chi_0 \to_C \chi_1 \to_C \cdots \to_C \chi_n = \varphi$ as a derivation of the statement that $\varphi \in Clos(\psi)$.

Remark 2.32 The final example of Remark 2.12 shows that the closure of a non-tidy formula may not even be defined — unless we work with alphabetical variants. We will come back to this point later. $\lhd$

The following example will be instructive for understanding the concept of closure, and its relation with subformulas.

Example 2.33 Consider the following formulas:

$$\begin{array}{lcl}
\xi_1 & := & \mu x_1 \nu x_2 \mu x_3.\big((x_1 \vee x_2 \vee x_3) \wedge \Box(x_1 \vee x_2 \vee x_3)\big) \\
\xi_2 & := & \nu x_2 \mu x_3.\big((\xi_1 \vee x_2 \vee x_3) \wedge \Box(\xi_1 \vee x_2 \vee x_3)\big) \\
\xi_3 & := & \mu x_3.\big((\xi_1 \vee \xi_2 \vee x_3) \wedge \Box(\xi_1 \vee \xi_2 \vee x_3)\big) \\
\xi_4 & := & \big((\xi_1 \vee \xi_2 \vee \xi_3) \wedge \Box(\xi_1 \vee \xi_2 \vee \xi_3)\big) \\
\alpha & := & \xi_1 \vee \xi_2 \vee \xi_3 \\
\beta & := & \xi_1 \vee \xi_2,
\end{array}$$

and let Φ be the set $\Phi := \{\xi_1, \xi_2, \xi_3, \xi_4, \Box\alpha, \alpha, \beta\}$.

For $i = 1, 2, 3$, the formula ξ_{i+1} is the unfolding of the formula ξ_i. Thus we find $Clos(\xi_1) = \Phi$; in fact, we have $Clos(\varphi) = \Phi$ for every formula $\varphi \in \Phi$. In Figure 1 we depict the *closure graph* of ξ_1.

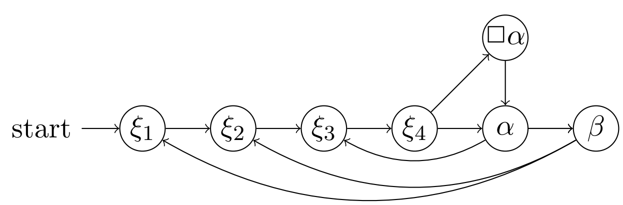


Figure 1: A closure graph

Observe that the formulas ξ_1, ξ_2, ξ_3 and ξ_4 are equivalent to one another, and hence also to α. Note too that the formula ξ_1 is the only clean formula in Φ. $\lhd$

It may not be immediately obvious from the definitions, but the closure set of a formula is always *finite*. We defer a proof of this proposition to the end of this section.

Proposition 2.34 *Let $\xi \in \mu\mathtt{ML}$ be some formula. Then the set $Clos(\xi)$ is finite.*

While Example 2.33 clearly shows that the unfolding of a clean formula will generally not be clean, tidyness is preserved.

Proposition 2.35 *Let $\xi \in \mu\mathtt{ML}$ be a tidy formula, and let φ be derived from ξ. Then*
1) $BV(\varphi) \subseteq BV(\xi)$ and $FV(\varphi) \subseteq FV(\xi)$;
2) φ is tidy;
3) if ψ is free for x in ξ then it is also free for x in φ.

Proof. The proofs of all three items proceed by a straightforward induction on the trace $\xi \twoheadrightarrow_C \varphi$. For instance, for the preservation of tidyness it suffices to prove that χ is tidy if $\heartsuit\chi$ is so (where $\heartsuit \in \{\Diamond, \Box\}$), that χ_0 and χ_1 are tidy if $\chi_0 \odot \chi_1$ is so (where $\odot \in \{\wedge, \vee\}$), and that the unfolding of a tidy formula is tidy again. The proofs of the first two claims are easy, and the third claim was stated in Proposition 2.13. QED

In many respects the closure and subformula maps behave in similar ways. In particular, we may also define an *evaluation game* using the closure set of a (tidy) formula. As we will see later on, this in fact motivates the choice of the number of elements in a formula's closure set as a suitable size measure. The winning condition of this evaluation game can be defined using the following observation, which in some sense is the analogon of Proposition 2.19.

Proposition 2.36 *Let $(\xi_n)_{n<\omega}$ be an infinite trace of tidy formulas. Then there is a unique fixpoint formula $\xi = \eta x.\chi$ which occurs infinitely often on the trace and is a subformula of ξ_n for cofinitely many n.*

▶ `Proof to be supplied.`

Definition 2.37 Let $\mathbb{S} = (S, R, V)$ be a Kripke model and let ξ be a tidy formula in $\mu\mathtt{ML}$. We define the *evaluation game* $\mathcal{E}^c(\xi, \mathbb{S})$ as the game (G, E, Ω) of which the board consists of the set $Clos(\xi) \times S$, and the game graph (i.e., the partitioning of $Clos(\xi) \times S$ into positions for the two players, together with the set $E(z)$ of admissible moves at each position), is given in Table 5.

Position	Player	Admissible moves
$(\varphi \vee \psi, s)$	$\exists$	$\{(\varphi, s), (\psi, s)\}$
$(\varphi \wedge \psi, s)$	$\forall$	$\{(\varphi, s), (\psi, s)\}$
$(\Diamond\varphi, s)$	$\exists$	$\{(\varphi, t) \mid sRt\}$
$(\Box\varphi, s)$	$\forall$	$\{(\varphi, t) \mid sRt\}$
(p, s) with $p \in FV(\xi)$ and $s \in V(p)$	$\forall$	$\varnothing$
(p, s) with $p \in FV(\xi)$ and $s \notin V(p)$	$\exists$	$\varnothing$
$(\overline{p}, s)$ with $p \in FV(\xi)$ and $s \in V(p)$	$\exists$	$\varnothing$
$(\overline{p}, s)$ with $p \in FV(\xi)$ and $s \notin V(p)$	$\forall$	$\varnothing$
$(\eta x.\varphi, s)$	-	$\{(\varphi[\eta x\, \varphi/x], s)\}$

Table 5: The closure evaluation game $\mathcal{E}^c(\xi, \mathbb{S})$

To define the winner of an infinite match $\Sigma = (\xi_n, s_n)_{n\in\omega}$, let $\xi = \eta x.\chi$ be the fixpoint formula, given by Proposition 2.36, that occurs infinitely often on the trace $(\xi_n)_{n\in\omega}$ and is a subformula of ξ_n for cofinitely many n. Then we declare $\exists$ ($\forall$) as the winner of Σ if $\eta = \nu$ (if $\eta = \mu$, respectively). $\lhd$

There are some noteworthy differences between subformulas and derived formulas as well. In particular, the subformula dag is clearly *acyclic* since any formula is longer than its subformulas; Example 2.33 clearly shows that this is not the case for the closure graph.

In the proposition below we see how the closure map interacts with various connectives and formula constructors of the μ-calculus.

Proposition 2.38 *Let χ and ξ be tidy formulas. Then the following hold:*

1) *if $\chi \trianglelefteq \xi$ is a literal then $\chi \in Clos(\xi)$;*
2) *if $\xi = \heartsuit\chi$, then $Clos(\xi) = \{\heartsuit\chi\} \cup Clos(\chi)$, where $\heartsuit \in \{\Diamond, \Box\}$;*
3) *if $\xi = \chi_0 \odot \chi_1$ then $Clos(\xi) = \{\chi_0 \odot \chi_1\} \cup Clos(\chi_0) \cup Clos(\chi_1)$, where $\odot \in \{\wedge, \vee\}$;*
4) *if $\xi = \chi[\psi/x]$ then $Clos(\xi) = \{\varphi[\psi/x] \mid \varphi \in Clos(\chi)\} \cup Clos(\psi)$, provided $x \in FV(\chi)$ and ψ is free for x in χ;*
5) *if $\xi = \eta x.\chi$ then $Clos(\xi) = \{\eta x.\chi\} \cup \{\varphi[\eta x.\chi/x] \mid \varphi \in Clos(\chi)\}$, where $\eta \in \{\mu, \nu\}$.*

Proof. Leaving the relatively easy proofs of the second and third claim to the reader, we first prove the fourth and fifth item, The first statement is an instance of Proposition 2.40, which we will prove later.

For the proof of 4), assume that $x \in FV(\chi)$ and that ψ is free for x in χ. By Proposition 2.35(3), the formula ψ is free for x in every $\varphi \in Clos(\chi)$. To prove the inclusion $\subseteq$ it suffices to show that the set $\{\varphi[\psi/x] \mid \varphi \in Clos(\chi)\} \cup Clos(\psi)$ has the required closure properties. This is easily verified, and so we omit the details.

For the opposite inclusion, we first show that

$$\varphi[\psi/x] \in Clos(\chi[\psi/x]), \text{ for all } \varphi \in Clos(\chi), \tag{11}$$

and we prove this by induction on the trace from ξ to χ. It is immediate by the definitions that $\chi[\psi/x] \in Clos(\chi[\psi/x])$, which takes care of the base case of this induction.

In the inductive step we distinguish three cases. First, assume that $\varphi \in Clos(\chi)$ because the formula $\heartsuit\varphi \in Clos(\chi)$, with $\heartsuit \in \{\Diamond, \Box\}$. Then by the inductive hypothesis we find $\heartsuit\varphi[\psi/x] = (\heartsuit\varphi)[\psi/x] \in Clos(\chi[\psi/x])$; but then we may immediately conclude that $\varphi[\psi/x] \in Clos(\chi[\psi/x])$ as well. The second case, where we assume that $\varphi \in Clos(\chi)$ because there is some formula $\varphi \odot \varphi'$ or $\varphi' \odot \varphi$ in $Clos(\chi)$ (with $\odot \in \{\wedge, \vee\}$), is dealt with in a similar way.

In the third case, we assume that $\varphi \in Clos(\chi)$ is of the form $\varphi = \rho[\langle y.\rho/y]$, with $\langle \in \{\mu, \nu\}$ and $\langle y.\rho \in Clos(\chi)$. Then inductively we may assume that $(\langle y.\rho)[\psi/x] \in Clos(\chi[\psi/x])$. Now we make a case distinction: if $x = y$ we find that $(\langle y.\rho)[\psi/x] = \langle y.\rho$, while at the same time we have $\varphi[\psi/x] = \rho[\langle y.\rho/y][\psi/x] = \rho[\langle y.\rho/y]$, so that it follows by the closure properties that $\varphi[\psi/x] \in Clos(\chi)$ indeed. If, on the other hand, x and y are distinct variables, then we find $(\langle y.\rho)[\psi/x] = \langle y.\rho[\psi/x]$, and so it follows by the closure properties that the formula $(\rho[\psi/x])\big[\langle y.\rho[\psi/x]/y\big]$ belongs to $Clos(\chi[\psi/x])$. But since ψ is free for x in χ, the variable y is not free in ψ, and so a straightforward calculation shows that $(\rho[\psi/x])\big[\langle y.\rho[\psi/x]/y\big] = \rho[\langle y.\rho/y][\psi/x] = \varphi[\psi/x]$, and so we find that $\varphi[\psi/x] \in Clos(\chi[\psi/x])$ in this case as well.

Now we turn to claim 5) of the proposition. First observe that by Proposition 2.13(1), the formula $\eta x.\chi$ is free for x in χ, so that we may apply part 4) without any problem. For the proof of the inclusion '$\subseteq$' it suffices to show that the set $\{\eta x.\chi\} \cup \{\varphi[\eta x.\chi/x] \mid \varphi \in$

$Clos(\chi)\}$ has the right closure properties, which is easy. For the opposite inclusion '$\supseteq$', it is immediate by the definitions that $Clos(\eta x.\chi) = \{\eta x.\chi\} \cup Clos(\chi[\eta x.\chi/x])$. But we saw in Proposition 2.13 that the formula $\eta x.\chi$ is free for x in χ. It then follows by 4) that $Clos(\eta x.\chi) = \{\eta x.\chi\} \cup \{\varphi[\eta x.\chi/x] \mid \varphi \in Clos(\chi)\} \cup Clos(\eta x.\chi)$, whence the inclusion '$\supseteq$' is immediate. QED

Subformulas and derived formulas

We now have a closer look at the relation between the sets $Sfor(\xi)$ and $Clos(\xi)$. Our first observation concerns the question, which subformulas of a formula also belong to its closure. This brings us to the notion of a *free* subformula.

Definition 2.39 Let φ and ψ be μ-calculus formulas. We say that φ is a *free* subformula of ψ, notation: $\varphi \trianglelefteq_f \psi$, if $\psi = \psi'[\varphi/x]$ for some formula ψ' such that $x \in FV(\psi')$ and φ is free for x in ψ'. $\triangleleft$

Note that in particular all literals occurring in ψ are free subformulas of ψ. The following characterisation is useful. Recall that we write $\varphi \twoheadrightarrow_C \psi$ if $\psi \in Clos(\varphi)$, or equivalently, if there is a trace (possibly of length zero) from φ to ψ.

Proposition 2.40 *Let φ and ψ be μ-calculus formulas. If ψ is tidy, then the following are equivalent:*

1) $\varphi \trianglelefteq_f \psi$;
2) $\varphi \trianglelefteq \psi$ and $FV(\varphi) \cap BV(\psi) = \varnothing$;
3) $\varphi \trianglelefteq \psi$ and $\psi \twoheadrightarrow_C \varphi$.

Proof. We will prove the equivalence of the statements 1) - 3) to a fourth statement, viz.:
4) there is a $\triangleleft_0$-chain $\varphi = \chi_0 \triangleleft_0 \chi_1 \triangleleft_0 \cdots \triangleleft_0 \chi_n = \psi$, such that no χ_i has the form $\chi_i = \eta y.\rho_i$ with $y \in FV(\varphi)$.

For the implication 1) $\Rightarrow$ 4), assume that $\varphi \trianglelefteq_f \psi$, then by definition ψ is of the form $\psi'[\varphi/x]$ where $x \in FV(\psi')$ and φ is free for x in ψ'. But if $x \in FV(\psi)$, then it is easy to see that there is a $\triangleleft_0$-chain $x = \chi'_0 \triangleleft_0 \chi'_1 \triangleleft_0 \cdots \triangleleft_0 \chi'_n = \psi'$ such that no χ'_i is of the form $\chi'_i = \langle x.\rho'$. Assume for contradiction that one of the formulas χ'_i is of the form $\chi_i = \eta y.\rho_i$ where $y \in FV(\varphi)$. Since φ is free for x in ψ' this would mean that there is a formula of the form $\langle x.\chi$ with $\eta y.\rho_i \trianglelefteq \langle x.\chi \trianglelefteq \psi'$. However, the only candidates for this would be the formulas χ'_j with $j > i$, and we just saw that these are not of the shape $\langle x.\rho'$. This provides the desired contradiction.

For the opposite implication 4) $\Rightarrow$ 1), assume that there is a $\triangleleft_0$-chain $\varphi = \chi_0 \triangleleft_0 \chi_1 \triangleleft_0 \cdots \triangleleft_0 \chi_n = \psi$ as in the formulation of 4). One may then show by a straightforward induction that $\varphi \trianglelefteq_f \chi_i$, for all $i \geq 0$.

For the implication 2) $\Rightarrow$ 4), assume that $\varphi \trianglelefteq \psi$ and $FV(\varphi) \cap BV(\psi) = \varnothing$. It follows from $\varphi \trianglelefteq \psi$ that there is a $\triangleleft_0$-chain $\varphi = \chi_0 \triangleleft_0 \chi_1 \triangleleft_0 \cdots \triangleleft_0 \chi_n = \psi$. Now suppose for contradiction that one of the formulas χ_i would be of the form $\chi_i = \eta y.\rho_i$ with $y \in FV(\varphi)$. Then we would find $y \in FV(\varphi) \cap BV(\psi)$, contradicting the assumption that $FV(\varphi) \cap BV(\psi) = \varnothing$.

In order to prove the implication 4) $\Rightarrow$ 3), it suffices to show, for any n, that if $(\chi_i)_{0 \leq i \leq n}$ is an $\lhd_0$-chain of length $n+1$ such that no χ_i has the form $\chi_i = \eta y.\rho_i$ with $y \in FV(\chi_0)$, then $\chi_n \twoheadrightarrow_C \chi_0$. We will prove this claim by induction on n. Clearly the case where $n = 0$ is trivial.

For the inductive step we consider a chain

$$\chi_0 \lhd_0 \chi_1 \lhd_0 \cdots \lhd_0 \chi_n \lhd_0 \chi_{n+1}$$

such that no χ_i has the form $\chi_i = \eta y.\rho_i$ with $y \in FV(\chi_0)$, and we make a case distinction as to the nature of χ_{n+1}. Clearly χ_{n+1} cannot be an atomic formula.

If χ_{n+1} is of the form $\rho_0 \odot \rho_1$ with $\odot \in \{\wedge, \vee\}$, then since $\chi_n \lhd_0 \chi_{n+1}$, the first formula must be of the form $\chi_n = \rho_i$ with $i \in \{0, 1\}$. But since it follows by the induction hypothesis that $\chi_n \twoheadrightarrow_C \chi_0$, we obtain from $\chi_{n+1} \rightarrow_C \chi_n$ that $\chi_{n+1} \twoheadrightarrow_C \chi_0$ as required. The case where χ_{n+1} is of the form $\heartsuit\rho$ with $\heartsuit \in \{\Diamond, \Box\}$ is handled similarly.

This leaves the case where $\chi_{n+1} = \lambda y.\rho$ is a fixpoint formula. Then since $\chi_n \lhd_0 \chi_{n+1}$ it must be the case that $\chi_n = \rho$. Furthermore, it follows from the assumption in 4) that $y \notin FV(\chi_0)$. From this it is not so hard to see that

$$\chi_0 \lhd_0 \chi_1[\chi_{n+1}/y] \lhd_0 \cdots \lhd_0 \chi_n[\chi_{n+1}/y]$$

is a $\lhd_0$-chain to which the induction hypothesis applies. It follows that $\chi_n[\chi_{n+1}/y] \twoheadrightarrow_C \chi_0$. From this and the observation that $\chi_{n+1} \rightarrow_C \chi_n[\chi_{n+1}/y]$ we find that $\chi_{n+1} \twoheadrightarrow_C \chi_0$ indeed. This finishes the proof of the implication 4) $\Rightarrow$ 3).

Finally, it follows from Proposition 2.35(1) that $\psi \twoheadrightarrow_C \varphi$ implies $FV(\varphi) \cap BV(\psi) \subseteq FV(\psi) \cap FV(\psi) = \varnothing$. From this the implication 3) $\Rightarrow$ 2) is immediate. QED

As a nice application of the notion of a *free subformula*, the following proposition states that under some mild conditions, the substitution operation $[\xi/x]$ is in fact injective. We leave the proof of this proposition as an exercise to the reader.

Proposition 2.41 *Let φ_0, φ_1 and ξ be formulas such that ξ is free for x in both φ_0 and φ_1, and not a free subformula of either φ_i. Then*

$$\varphi_0[\xi/x] = \varphi_1[\xi/x] \text{ implies } \varphi_0 = \varphi_1. \tag{12}$$

The most important observation here concerns the existence of a surjective map from *Sfor*(ξ) to *Clos*(ξ), at least for a clean formula ξ. Recall that, given a clean formula ξ, we define the *dependency order* $<_\xi$ on the bound variables of ξ as the least strict partial order such that $x <_\xi y$ if $\delta_x \lhd \delta_y$ and $y \trianglelefteqslant \delta_x$.

Definition 2.42 Writing $BV(\xi) = \{x_1, \ldots, x_n\}$, where we may assume that $i < j$ if $x_i <_\xi x_j$, we define the *expansion* $\exp_\xi(\varphi)$ of a subformula φ of ξ as:

$$\exp_\xi(\varphi) := \varphi[\eta_{x_1} x_1.\delta_{x_1}/x_1] \ldots [\eta_{x_n} x_n.\delta_{x_n}/x_n].$$

That is, we substitute first x_1 by $\eta_{x_1} x_1.\delta_{x_1}$ in φ; in the resulting formula, we substitute x_2 by $\eta_{x_2} x_2.\delta_{x_2}$, etc. If no confusion is likely we write $\exp(\varphi)$ instead of $\exp_\xi(\varphi)$. A proposition letter p is *active in* φ if p occurs in δ_y for some $y >_\xi x$, or equivalently, if p occurs in $\exp_\xi(\varphi)$. $\lhd$

Without proof we mention the following result.

Proposition 2.43 *Let $\xi \in \mu\mathtt{ML}$ be a formula and $\mathbb{S}$ a pointed Kripke structure. Then for all subformulas $\varphi \trianglelefteq \xi$ and all states s in $\mathbb{S}$ we have*

$$(\varphi, s) \in \mathrm{Win}_{\exists}(\mathcal{E}(\xi, \mathbb{S})) \textit{ iff } \mathbb{S}, s \Vdash_g \exp_{\xi}(\varphi).$$

Proposition 2.44 *Let ξ be a clean $\mu\mathtt{ML}$-formula. Then*

$$Clos(\xi) = \{\exp_{\xi}(\varphi) \mid \varphi \trianglelefteq \xi\}. \tag{13}$$

Proof. For the time being we confine ourselves to a brief sketch. For the inclusion $\subseteq$ it suffices to show that the set $\{\exp_{\xi}(\varphi) \mid \varphi \trianglelefteq \xi\}$ has the relevant closure properties. This is a fairly routine proof. For the opposite inclusion it suffices to prove that $\exp_{\xi}(\varphi) \in Clos(\xi)$, for every $\varphi \in Sfor(\xi)$, which can be done by a straightforward induction. QED

The size of a formula

As an immediate corollary of Proposition 2.44 we find that the closure set of a μ-calculus formula is always *finite* — this proves Proposition 2.34. We will see further on that in fact, the number of formulas that can be derived from a formula may be *exponentially smaller* than its number of subformulas, and that the first number is a more suitable size measure than the latter.

Definition 2.45 The *size* $|\xi|$ of a formula ξ is given by

$$|\xi| := |Clos(\xi)|,$$

i.e., it is defined as the number of formulas that are derived from ξ. $\triangleleft$

2.6 Alternation depth

After size, the most important complexity measure of modal μ-calculus formulas concerns the degree of nesting of least- and greatest fixpoint operators in the syntax tree (or dag) of the formula. Intuitively, the *alternation depth* of a formula ξ will be defined as the length of a maximal chain of nested, alternating fixpoint operators. As in the case of size, there is more than one reasonable way to make this intuition precise

As a first example, consider the formula

$$\xi_1 = \mu x.(\nu y.p \wedge \Box y) \vee \Diamond x,$$

expressing the reachability of some state from which only p-states will be reachable. Clearly this formula witnesses a ν-operator in the scope of a μ-operator, and in the most straightforward approach one might indeed take this as nesting, and define the (simple) alternation depth of the formula ξ_1 as 2. However, a closer inspection of the formula ξ_1 reveals that, since the variable x does not occur in the subformula $\nu y.p \wedge \Box y$, the latter subformula does not really depend on x. This is different in the following example:

$$\xi_2 = \nu x.\mu y.(p \wedge \Diamond x) \vee \Diamond y,$$

stating the existence of a path on which p is true infinitely often. Here the variable x does occur in the subformula $\mu y.(p \wedge \Diamond x) \vee \Diamond y$; that is, ξ_2 contains a 'real' ν/μ-chain of fixpoint operators. In the definition of alternation depth ad that we shall adopt, we will see that $ad(\xi_2) = 2$ but $ad(\xi_1) = 1$.

The formal definition of alternation depth involves inductively defined formula collections Θ^η_n, where $\eta \in \{\mu, \nu\}$ and n is a natural number. Intuitively, the class Θ^η_n consists of those μ-calculus formulas where n bounds the length of any alternating nesting of fixpoint operators of which the most significant formula is an η-formula. We will make this intuition more precise further on.

For the next definition, recall our notation $\overline{\mu} = \nu$, $\overline{\nu} = \mu$.

Definition 2.46 By natural induction we define classes $\Theta^\mu_n, \Theta^\nu_n$ of μ-calculus formulas. With $\eta, \lambda \in \{\mu, \nu\}$ arbitrary, we set:

1. all atomic formulas belong to Θ^η_0;
2. if $\varphi_0, \varphi_1 \in \Theta^\eta_n$, then $\varphi_0 \vee \varphi_1, \varphi_0 \wedge \varphi_1, \Diamond\varphi_0, \Box\varphi_0 \in \Theta^\eta_n$;
3. if $\varphi \in \Theta^\eta_n$ then $\overline{\eta}x.\varphi \in \Theta^\eta_n$;
4. if $\varphi(x), \psi \in \Theta^\eta_n$, then $\varphi[\psi/x] \in \Theta^\eta_n$, provided that ψ is free for x in φ;
5. all formulas in Θ^λ_n belong to Θ^η_{n+1}.

The *alternation depth* $ad(\xi)$ of a formula ξ is defined as the least n such that $\xi \in \Theta^\mu_n \cap \Theta^\nu_n$.

A formula is *alternation free* if it has alternation depth 1. $\lhd$

Example 2.47 Observe that the basic modal (i.e., fixpoint-free) formulas are exactly the ones with alternation depth zero. Formulas that use μ-operators or ν-operators, but not both, have alternation depth 1. For example, observe that $\mu x.p \vee x$ belongs to Θ^ν_0 but not to Θ^μ_0: none of the clauses in Definition 2.46 is applicable. On the other hand, using clause (5) it is easy to see that $\mu x.p \vee x \in \Theta^\nu_1 \cap \Theta^\mu_1$, from which it is immediate that $ad(\mu x.p \vee x) = 1$.

Consider the formula $\xi_1 = \mu x.(\nu y.p \wedge \Box y) \wedge \Diamond x$. Taking a fresh variable q, we find $\mu x.q \wedge \Diamond x \in \Theta^\nu_0 \subseteq \Theta^\nu_1$ and $\nu y.p \wedge \Box y \in \Theta^\mu_0 \subseteq \Theta^\nu_1$, so that by the substitution rule we have $\xi_1 = (\mu x.q \wedge \Diamond x)[\nu y.p \wedge \Box y/q] \in \Theta^\nu_1$. Similarly we may show that $\xi_1 \in \Theta^\mu_1$, so that ξ_1 has alternation depth 1.

The formula $\xi_2 = \nu x.\mu y.(p \wedge \Diamond x) \vee \Diamond y$ is of higher complexity. It is clear that the formula $\mu y.(p \wedge \Diamond x) \vee \Diamond y$ belongs to Θ^ν_0 but not to Θ^μ_0. From this it follows that ξ_2 belongs to Θ^μ_1 but there is no way to place it in Θ^ν_1. Hence we find that $ad(\xi_2) = 2$.

As a third example, consider the formula

$$\xi_3 = \mu x.\nu y.(\Box y \wedge \mu z.(\Diamond x \vee z)).$$

This formula looks like a $\mu/\nu/\mu$-formula, in the sense that it contains a nested fixpoint chain $\mu x/\nu y/\mu z$. However, the variable y does not occur in the subformula $\mu z.(\Diamond x \vee z)$, and so we may in fact consider ξ_3 as a μ/ν-formula. Formally, we observe that $\mu z.\Diamond x \vee z \in \Theta_0^\nu \subseteq \Theta_1^\nu$ and $\nu z.\Box y \wedge p \in \Theta_0^\mu \subseteq \Theta_1^\nu$; from this it follows by the substitution rule that the formula $\nu y.(\Box y \wedge \mu z.(\Diamond x \vee z))$ belongs to the set Θ_1^ν as well; from this it easily follows that $\xi_3 \in \Theta_1^\nu$. It is not hard to show that $\xi_3 \notin \Theta_1^\mu$, so that we find $ad(\xi_3) = 2$. $\lhd$

In the propositions below we make some observations on the sets Θ_n^η and on the notion of alternation depth. First we show that each class Θ_n^μ is closed under subformulas and derived formulas.

Proposition 2.48 *Let Let ξ and φ be μ-calculus formulas.*
1) If $\varphi \trianglelefteq \xi$ and $\xi \in \Theta_n^\eta$ then $\varphi \in \Theta_n^\eta$.
2) If $\xi \twoheadrightarrow_C \varphi$ and $\xi \in \Theta_n^\eta$ then $\varphi \in \Theta_n^\eta$.

Proof. We prove the statement in part 1) by induction on the derivation of $\xi \in \Theta_n^\eta$. In the base case of this induction we have that $n = 0$ and ξ is an atomic formula. But then obviously all subformulas of ξ are atomic as well and thus belong to Θ_n^η.

In the induction step of the proof it holds that $n > 0$; we make a case distinction as to the applicable clause of Definition 2.46.

In case $\xi \in \Theta_n^\eta$ because of clause (2) in Definition 2.46, we make a further case distinction as to the syntactic shape of ξ. First assume that ξ is a conjunction, say, $\xi = \xi_0 \wedge \xi_1$, with $\xi_0, \xi_1 \in \Theta_n^\eta$. Now consider an arbitrary subformula φ of ξ; it is not hard to see that either $\varphi = \xi$ or $\varphi \trianglelefteq \xi_i$ for some $i \in \{0, 1\}$. In the first case we are done, by assumption that $\xi \in \Theta_n^\eta$; in the second case, we find $\varphi \in \Theta_n^\eta$ as an immediate consequence of the induction hypothesis. The cases where ξ is a disjunction, or a formula of the form $\Box\psi$ or $\Diamond\psi$ are treated in a similar way.

If $\xi \in \Theta_n^\eta$ because of clause (3) of the definition, then ξ must be of the form $\xi = \eta x.\chi$, with $\chi \in \Theta_n^\eta$. We proceed in a way similar to the previous case: any subformula $\varphi \trianglelefteq \xi$ is either equal to ξ (in which case we are done by assumption), or a subformula of χ, in which we are done by one application of the induction hypothesis.

In the case of clause (4), assume that ξ is of the form $\chi[\psi/x]$, where ψ is free for x in χ, and χ and ψ are in Θ_n^η. Then by the induction hypothesis all subformulas of χ and ψ belong to Θ_n^η as well. Now consider an arbitrary subformula φ of ξ; it is easy to see that either $\varphi \trianglelefteq \chi$, $\varphi \trianglelefteq \psi$ or else φ is of the form $\varphi = \varphi'[\psi/x]$ where $\varphi' \trianglelefteq \chi$. In either case it is straightforward to prove that $\varphi \in \Theta_n^\eta$, as required.

Finally, in case ξ is in Θ_n^η because of clause (5), it belongs to Θ_{n-1}^λ for some $\lambda \in \{\mu, \nu\}$. Then by induction hypothesis all subformulas of ξ belong to Θ_{n-1}^λ. We may then apply the same clause (5) to see that any such φ also belongs to the set Θ_n^η.

To prove part 2), it suffices to show that the class Θ_n^η is closed under unfoldings, since by part 1) we already know it to be closed under subformulas. So assume that $\lambda x.\chi \in \Theta_n^\eta$ for some n and $\lambda \in \{\mu, \nu\}$. Because $\chi \trianglelefteq \eta x.\chi$ it follows from part 1) that $\chi \in \Theta_n^\lambda$. But then we may apply clause (4) from Definition 2.46 and conclude that $\chi[\eta.\chi/x] \in \Theta_n^\lambda$. QED

As an immediate corollary of Proposition 2.48 we find the following.

Proposition 2.49 *Let ξ and χ be μ-calculus formulas. Then*

1) if $\chi \in Sfor(\xi)$ then $ad(\chi) \leq ad(\xi)$;
2) if $\chi \in Clos(\xi)$ then $ad(\chi) \leq ad(\xi)$.

In the case of a *clean* formula there is a simple characterisation of alternation depth, making precise the intuition about alternating chains, in terms of the formula's dependency order on the bound variables.

Definition 2.50 Let $\xi \in \mu\mathtt{ML}$ be a clean formula. A *dependency chain* in ξ of *length* d is a sequence $\overline{x} = x_1 \cdots x_d$ such that $x_1 <_\xi x_2 \cdots <_\xi x_d$; such a chain is *alternating* if x_i and x_{i+1} have different parity, for every $i < d$. For $\eta \in \{\mu, \nu\}$, we call an alternating dependency chain $x_1 \cdots x_d$ an *η-chain* if x_d is an η-variable, and we let $d_\eta(\xi)$ denote the length of the longest η-chain in ξ; we write $d_\eta(\xi) = 0$ if ξ has no such chains. $\lhd$

Proposition 2.51 *Let ξ be a clean formula. Then for any $k \in \omega$ and $\eta \in \{\mu, \nu\}$ we have*

$$\xi \in \Theta^\eta_k \text{ iff } d_\eta(\xi) \leq k, \tag{14}$$

As a corollary, the alternation depth of ξ is equal to the length of its longest alternating dependency chain.

One of the key insights in the proof of this Proposition is that, with ψ free for x in φ, any dependency chain in $\varphi[\psi/x]$ originates entirely from either φ or ψ. Recall from Definition 2.2 that we write $\overline{\mu} = \nu$ and $\overline{\nu} = \mu$.

Proof. We prove the implication from left to right in (14) by induction on the derivation that $\xi \in \Theta^\eta_k$. In the base step of this induction (corresponding to clause (1) in the definition of alternation depth) ξ is atomic, so that we immediately find $d_\eta(\xi) = 0$ as required.

In the induction step of the proof, we make a case distinction as to the last applied clause in the derivation of $\xi \in \Theta^\eta_k$, and we leave the (easy) cases, where this clause was either (2) or (3), for the reader.

Suppose then that $\xi \in \Theta^\eta_k$ on the basis of clause (4). In this case we find that $\xi = \xi'[\psi/z]$ for some formulas ξ', ψ such that ψ is free for z in ξ' and $\xi', \psi \in \Theta^\eta_k$. By the 'key insight' mentioned right after the formulation of the Proposition, any η-chain in the formula ξ is a η-chain in either ξ' or ψ. But then by the induction hypothesis it follows that the length of any such chain must be bounded by k.

Finally, consider the case where $\xi \in \Theta^\eta_k$ on the basis of clause (5). We make a further case distinction. If $\xi \in \Theta^\eta_{k-1}$, then by the induction hypothesis we may conclude that $d_\eta(\xi) \leq k-1$, and from this it is immediate that $d_\eta(\xi) \leq k$. If, on the other hand, $\xi \in \Theta^{\overline{\eta}}_{k-1}$ then the induction hypothesis yields $d_{\overline{\eta}}(\xi) \leq k-1$. But since $d_\eta(\xi) \leq d_{\overline{\eta}}(\xi) + 1$ we obtain $d_\eta(\xi) \leq k$ indeed.

The opposite, right-to-left, implication in (14) is proved by induction on k. In the base step of this induction we have $d_\eta(\xi) = 0$, which means that ξ has no η-variables; from this it is easy to derive that $\xi \in \Theta^\eta_0$.

For the induction step, we assume as our induction hypothesis that (14) holds for $k \in \omega$, and we set out to prove the same statement for $k+1$ and an arbitrary $\eta \in \{\mu, \nu\}$:

$$\text{if } d_\eta(\xi) \leq k+1 \text{ then } \xi \in \Theta^\eta_{k+1}. \tag{15}$$

We will prove (15) by an 'inner' induction on the length of ξ. The base step of this inner induction is easy to deal with: if $|\xi| = 1$ then ξ must be atomic so that certainly $\xi \in \Theta^\eta_{k+1}$.

In the induction step we are considering a formula ξ with $|\xi| > 1$. Assume that $d_\eta(\xi) \leq k+1$. We make a case distinction as to the shape of ξ. The only case of interest is where ξ is a fixpoint formula, say, $\xi = \eta x.\chi$ or $\xi = \overline{\eta}x.\chi$. If $\xi = \overline{\eta}x.\chi$, then obviously we have $d_\eta(\xi) = \delta_\eta(\chi)$, so by the inner induction hypothesis we find $\chi \in \Theta^\eta_{k+1}$. From this we immediately derive that $\xi = \overline{\eta}x.\chi \in \Theta^\eta_{k+1}$ as well.

Alternatively, if $\xi = \eta x.\chi$, we split further into cases: If χ has an $\overline{\eta}$-chain $y_1 \cdots y_{k+1}$ of length $k+1$, then obviously we have $x \notin FV(\delta_{k+1})$ (where we write δ_{k+1} instead of $\delta_{y_{k+1}}$), for otherwise we would get $x >_\xi y_{k+1}$, so that we could add x to the $\overline{\eta}$-chain $y_1 \cdots y_{k+1}$ and obtain an η-chain $y_1 \cdots y_{k+1}x$ of length $k+2$. But if $x \notin FV(\delta_{k+1})$ we may take some fresh variable z and write $\xi = \xi'[\overline{\eta}y_{k+1}.\delta_{k+1}/z]$ for some formula ξ' where the formula $\overline{\eta}y_{k+1}.\delta_{k+1}$ is free for z. By our inner induction hypothesis we find that both ξ' and $\eta y_{k+1}.\delta_{k+1}$ belong to Θ^η_{k+1}. But then by clause (4) of Definition 2.46 the formula ξ also belongs to the set Θ^η_{k+1}.

If, on the other hand, χ has *no* $\overline{\eta}$-chain of length $k+1$, then we clearly have $d_{\overline{\eta}}(\chi) \leq k$. Using the outer induction hypothesis we infer $\chi \in \Theta^{\overline{\eta}}_k$, and so by clause (3) of Definition 2.46 we also find $\xi = \eta x.\chi \in \Theta^{\overline{\eta}}_k$. Finally then, clause (5) gives $\xi \in \Theta^\eta_{k+1}$. QED

One may prove a similar (but somewhat more involved) characterisation in the wider setting of tidy formulas, as we will see further on.

Notes

The modal μ-calculus was introduced by D. Kozen [15]. Its game-theoretical semantics goes back to at least Emerson & Jutla [11] (who use alternating automata as an intermediate step). As far as we are aware, the bisimulation invariance theorem, with the associated tree model property, is a folklore result. The bounded tree model property is due to Kozen & Parikh [17].

There are various ways to make the notion of alternation depth precise; we work with the most widely used definition, which originates with Niwiński [22].

▶ `More notes to be supplied.`

Exercises

Exercise 2.1 Express in words the meaning of the following μ-calculus formula:

$$\nu x.\mu y.(p \wedge \Box x) \vee (\overline{p} \wedge \Box y).$$

Exercise 2.2 (defining modal μ-formulas) Give a modal μ-formula $\varphi(p,q)$ such that for all transition systems $\mathbb{S}$, and all states s_0 in $\mathbb{S}$:

$$\mathbb{S}, s_0 \Vdash_g \varphi(p,q) \quad \text{iff} \quad \begin{array}{l}\text{there is a path } s_0 R s_1 \ldots R s_n \ (n \geq 0) \text{ such that } \mathbb{S}, s_n \Vdash_g p \\ \text{and } \mathbb{S}, s_i \Vdash_g q \text{ for all } i \text{ with } 0 \leq i < n.\end{array}$$

Exercise 2.3 (characterizing winning strategies)
A *board* is a structure $\mathbb{B} = \langle B_0, B_1, E\rangle$ such that $B_0 \cap B_1 = \varnothing$ and $E \subseteq B^2$, where $B = B_0 \uplus B_1$ is a set of objects called *positions*. A *match* on $\mathbb{B}$ consists of the *players* 0 and 1 moving a token from one position to another, following the edge relation E. Player i is supposed to move the token when it is situated on a position in B_i. Suppose in addition that B is also partitioned into green and red positions, $B = G \uplus R$.

We will use a modal language to describe this structure, with the modalities being interpreted by the edge relation E, the proposition letter p_0 and r referring to the positions belonging to player 0, and the red positions, respectively. That is, $V(p_0) = B_0$ and $V(r) = R$.

(a) Consider the game where player 0 wins as soon as the token reaches a green position. (That is, all infinite matches are won by player 1. Player 0 wins if player 1 gets stuck, or if the token reaches a green position; player 1 wins a finite match if player 0 gets stuck.) Show that the formula $\varphi_a = \mu x.\overline{r} \vee (p_0 \wedge \Diamond x) \vee (\overline{p}_0 \wedge \Box x)$ characterizes the winning positions for player 0 in this game, in the sense that for any position $b \in B$, we have

$$\mathbb{B}, V, b \Vdash_g \varphi \text{ iff } \text{ player 0 has a w.s. in the game starting at position } b.$$

(b) Now consider the game where player 0 wins if she manages to reach a green position *infinitely often*. (More precisely, infinite matches are won by 0 iff a green position is reached infinitely often; finite matches are lost by a player is he/she gets stuck.) Give a formula φ_b that characterizes the winning positions in this game.

Exercise 2.4 (characterizing fairness) Let $\mathsf{D} = \{a, b\}$ be the set of atomic actions, and consider the following formula ξ, with subformulas as indicated:

$$\xi = \nu x.\mu y.\nu z. \overbrace{\underbrace{\Box_a x}_{\alpha_1} \wedge \underbrace{(\Box_a \bot \vee \Box_b y)}_{\alpha_2} \wedge \underbrace{\Box_b z}_{\alpha_3}}^{\delta}$$

Fix an LTS $\mathbb{S} = (S, R_a, R_b, V)$. We say that the transition a is *enabled* at state s of $\mathbb{S}$ if $\mathbb{S}, s \Vdash_g \Diamond_a \top$.

Show that ξ expresses some kind of *fairness* condition, i.e., the absence of a path starting at s on which a is enabled infinitely often, but executed only finitely often. More precisely, prove that $\mathbb{S}, s \Vdash_g \xi$ iff there is *no* path of the form $s_0 \xrightarrow{d_0} s_1 \xrightarrow{d_1} s_2 \cdots$ such that $s = s_0$, $d_i \in \{a, b\}$ for all i, a is enabled at s_i for infinitely many i, but $d_i = a$ for only finitely many i.

Exercise 2.5 (filtration) Recall that, given a finite and subformula closed set of formulas Σ and a model $\mathbb{S} = (S, R, V)$, we say that a model $\mathbb{S}' = (S', R', V')$ is a *filtration* of $\mathbb{S}$ through Σ if there is a surjective map $f : S \to S'$ such that:

a) for all proposition letters $p \in \Sigma$: $u \in V(p)$ iff $f(u) \in V'(p)$.

b) uRv implies $f(u)R'f(v)$

c) if $\Diamond\varphi \in \Sigma$ and $f(u)R'f(v)$, then $\mathbb{S}, v \Vdash_g \varphi$ implies $\mathbb{S}, u \Vdash_g \Diamond\varphi$

d) $f(u) = f(v)$ if and only if u and v satisfy precisely the same formulas in Σ.

Say that a formula ξ of the μ-calculus *admits filtration* if, for every model $\mathbb{S}$, there is a finite set of formulas Σ containing ξ, and a filtration $\mathbb{S}'$ of $\mathbb{S}$ through Σ such that $\mathbb{S}', f(s) \Vdash_g \varphi$ iff $\mathbb{S}, s \Vdash_g \varphi$, for each s in $\mathbb{S}$ and each $\varphi \in \Sigma$.

Prove that the formula $\mu x.\Box x$ does not admit filtration.

Exercise 2.6 We write $\varphi \models \psi$ to denote that ψ is a *local consequence* of φ, that is, if for all pointed Kripke models $(\mathbb{S}, s)$ it holds that $\mathbb{S}, s \Vdash_g \varphi$ implies $\mathbb{S}, s \Vdash_g \psi$.

(a) Show that $\mu x.\nu y.\, \alpha(x, y) \models \nu y.\mu x.\, \alpha(x, y)$, for all formulas α.

(b) Show that $\mu x.\mu y.\, \alpha(x, y) \equiv \mu y.\mu x.\, \alpha(x, y)$, for all formulas α.

(c) Show that $\mu x.(x \vee \gamma(x)) \wedge \delta(x) \models \mu x.\gamma(x) \wedge \delta(x)$, for all formulas γ, δ.

Exercise 2.7 (boolean μ-calculus) Show that the least and greatest fixpoint operators do not add expressive power to classical propositional logic, or, in other words, that the modality-free fragment of the modal μ-calculus is expressively equivalent to classical propositional logic. (Hint: use Exercise 2.6(c).)

Exercise 2.8 (co-induction) Let φ, ψ be any two clean formulas of the modal μ-calculus such that ψ is free for x in φ; it will also be convenient to assume that ψ is not a subformula of φ. Show by a game semantic argument that the following so-called 'co-induction principle' holds for greatest fixpoints: if $\psi \models \varphi[\psi/x]$, then $\psi \models \nu x.\varphi$ also. Here we write '$\models$' for the local consequence relation, as in Exercise 2.6.

Exercise 2.9 (injectivity of substitution) Prove Proposition 2.41.

3 Fixpoints

The game-theoretic semantics of the modal μ-calculus introduced in the previous chapter has some attractive characteristics. It is intuitive, relatively easy to understand, and, as we shall see further on, it can be used to prove some strong properties of the formalism. However, there are drawbacks as well. In particular, the game-theoretical semantics is not *compositional*; that is, the meaning of a formula is not defined in terms of the meanings of its subformulas. These shortcomings vanish in the *algebraic semantics* that we are about to introduce. In order to define this term, we first consider an example.

Example 3.1 Recall that in Example 2.1, we informally introduced the formula $\mu x.p \vee \Diamond_d x$ as the smallest fixpoint or solution of the 'equation' $x \equiv p \vee \Diamond_d x$.

To make this intuition more precise, we have to look at the formula $\delta = p \vee \Diamond_d x$ 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 $\delta^{\mathbb{S}}_x$ on the power set of S:

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

In our example we have

$$\delta^{\mathbb{S}}_x(X) = V(p) \cup \langle R\rangle(X).$$

Now we can make precise why $\mu x.p \vee \Diamond_d x$ is a fixpoint formula: its extension, the set $[\![\mu x.p \vee \Diamond_d x]\!]$, is a fixpoint of the map $\delta^{\mathbb{S}}_x$:

$$[\![\mu x.p \vee \Diamond_d x]\!] = V(p) \cup \langle R\rangle([\![\mu x.p \vee \Diamond_d x]\!]).$$

In fact, as we shall see in this chapter, the formulas $\mu x.p \vee \Diamond_d x$ and $\nu x.p \vee \Diamond_d x$ are such that their extensions are the *least* and *greatest* fixpoints of the map $\delta^{\mathbb{S}}_x$, respectively. $\lhd$

It is worthwhile to discuss the theory of fixpoint operators at a more general level than that of modal logic. Before we turn to the definition of the algebraic semantics of the modal μ-calculus, we first discuss the general fixpoint theory of monotone operations on complete lattices.

3.1 General fixpoint theory

Basics

In this chapter we assume some familiarity[2] with partial orders and lattices (see Appendix A).

Definition 3.2 Let $\mathbb{P}$ and $\mathbb{P}'$ be two partial orders and let $f : P \to P'$ be some map. Then f is called *monotone* or *order preserving* if $f(x) \leq' f(y)$ whenever $x \leq y$, and *antitone* or *order reversing* if $f(x) \geq' f(y)$ whenever $x \leq y$. $\lhd$

[2] Readers lacking this background may take abstract complete lattices to be concrete power set algebras.

Definition 3.3 Let $\mathbb{P} = \langle P, \leq \rangle$ be a 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 $p \leq f(p)$, 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). These least and greatest fixpoints may also be called *extremal fixpoints*. $\lhd$

The following theorem is a celebrated result in fixpoint theory.

Theorem 3.4 (Knaster-Tarski) *Let $\mathbb{C} = \langle C, \bigvee, \bigwedge \rangle$ be a complete lattice, and let $f : C \to C$ be monotone. Then f has both a least and a greatest fixpoint, and these are given as*

$$\mathrm{LFP}.f = \bigwedge \mathrm{PRE}(f), \tag{16}$$

$$\mathrm{GFP}.f = \bigvee \mathrm{POS}(f). \tag{17}$$

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 by monotonicity that $f(q) \leq f(x)$ for all $x \in \mathrm{PRE}(f)$, and hence by definition of prefixpoints, $f(q) \leq x$ for all $x \in \mathrm{PRE}(f)$. In other words, $f(q)$ is a lower bound of the set $\mathrm{PRE}(f)$. Hence, by definition of q as the *greatest* such lower bound, we find $f(q) \leq q$, that is, 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 — by monotonicity of f it immediately follows from $f(y) \leq y$ that $f(f(y)) \leq f(y)$. QED

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

Definition 3.5 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:

$$\begin{array}{rclcrcl}
f^0_\mu(c) & := & c, & \qquad & f^0_\nu(c) & := & c, \\
f^{\alpha+1}_\mu(c) & := & f(f^\alpha_\mu(c)) & & f^{\alpha+1}_\nu(c) & := & f(f^\alpha_\nu(c)), \\
f^\lambda_\mu(c) & := & \bigvee_{\alpha<\lambda} f^\alpha_\mu(c) & & f^\lambda_\nu(c) & := & \bigwedge_{\alpha<\lambda} f^\alpha_\nu(c),
\end{array}$$

where λ denotes an arbitrary limit ordinal. $\lhd$

Proposition 3.6 *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, $f^\alpha_\mu(\bot) \leq f^\beta_\mu(\bot)$ for all ordinals α and β such that $\alpha < \beta$.*

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

Given a set C, we let $|C|$ denote its cardinality or size.

Corollary 3.7 *Let $\mathbb{C} = \langle C, \bigvee, \bigwedge \rangle$ be a complete lattice, and let $f : C \to C$ be monotone. Then there is some α of size at most $|C|$ such that* LFP.$f = f^\alpha_\mu(\bot)$.

Proof. By Proposition 3.6, f is inductive, that is, $f^\alpha_\mu(\bot) \leq f^\beta_\mu(\bot)$ for all ordinals α and β such that $\alpha < \beta$. It follows from elementary set theory that there must be two ordinals α, β of size at most $|C|$ such that $f^\alpha_\mu(\bot) = f^\beta_\mu(\bot)$. From the definition of the approximations it then follows that there must be an ordinal α such that $f^\alpha_\mu(\bot) = f^{\alpha+1}_\mu(\bot)$, or, equivalently, $f^\alpha_\mu(\bot)$ is a fixpoint of f. To show that it is the *smallest* fixpoint, one may prove that $f^\beta_\mu(\bot) \leq \mathrm{LFP}.f$ for every ordinal β. This follows from a straightforward ordinal induction. QED

Definition 3.8 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 $f^\alpha_\mu(\bot) = f^{\alpha+1}_\mu(\bot)$ is called the *unfolding ordinal* of f. $\lhd$

3.2 Boolean algebras

In the special case that the complete lattice is in fact a (complete) *Boolean algebra*, there is more to be said.

Dual maps

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

Definition 3.9 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 $x \in B$. $\lhd$

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 3.10 Let $\mathbb{B} = \langle B, \bigvee, \bigwedge, - \rangle$ be a complete Boolean algebra. Given a map $f : B \to B$, the function $f^\partial : B \to B$ given by

$$f^\partial(b) := -f(-b).$$

is called the *(Boolean) dual* of f. $\lhd$

Proposition 3.11 *Let $\mathbb{B} = \langle B, \bigvee, \bigwedge, - \rangle$ be a complete Boolean algebra, and let $g : B \to B$ be monotone. Then g^∂ is monotone as well, $(g^\partial)^\partial = g$, and*

$$\begin{aligned} \mathrm{LFP}.g^\partial &= -\mathrm{GFP}.g, \\ \mathrm{GFP}.g^\partial &= -\mathrm{LFP}.g. \end{aligned}$$

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

First, note that by monotonicity of g^∂, the Knaster-Tarski theorem gives that

$$\mathrm{LFP}.g^\partial = \bigwedge \mathrm{PRE}(g^\partial).$$

But as a consequence of the definitions, we have that

$$b \in \mathrm{PRE}(g^\partial) \iff -b \in \mathrm{POS}(g).$$

From this it follows that

$$\begin{aligned} \mathrm{LFP}.g^\partial &= \bigwedge\{b \mid -b \in \mathrm{POS}(g)\} \\ &= \bigwedge\{-a \mid a \in \mathrm{POS}(g)\} \\ &= -\bigvee \mathrm{POS}(g) \\ &= -\mathrm{GFP}.g \end{aligned}$$

which finishes the proof of the Theorem. QED

Further on we will see that Proposition 3.11 allows us to define negation as an abbreviated operator in the modal μ-calculus.

Games

In case the Boolean algebra in question is in fact a *power set algebra*, a nice game-theoretic characterization of least and greatest fixpoint operators can be given.

Definition 3.12 Let S be some set and let $F : \wp(S) \to \wp(S)$ be a monotone operation. Consider the *unfolding games* $\mathcal{U}^\mu(F)$ and $\mathcal{U}^\nu(F)$. The positions and admissible moves of these two graph games are the same, see Table 6.

Position	Player	Admissible moves
$s \in S$	$\exists$	$\{A \in \wp(S) \mid s \in F(A)\}$
$A \in \wp(S)$	$\forall$	$A (= \{s \in S \mid s \in A\})$

Table 6: Unfolding games for $F : \wp(S) \to \wp(S)$

The *winning conditions* of finite matches are standard (the player that got stuck loses the match). The difference between $\mathcal{U}^\mu(F)$ and $\mathcal{U}^\nu(F)$ shows up in the winning conditions of infinite matches: $\exists$ wins the infinite matches of $\mathcal{U}^\nu(F)$, but $\forall$ those of $\mathcal{U}^\mu(F)$. $\lhd$

Example 3.13 In fact, we have already seen an example of the unfolding game $\mathcal{U}^\nu$ in the *bisimilarity game* of Definition 1.25. Given two Kripke models $\mathbb{S}$ and $\mathbb{S}'$, consider the map $F : \wp(S \times S')$ given by

$$F(Z) := \{(s, s') \in S \times S' \mid Z \text{ is a local bisimulation for } s \text{ and } s'\},$$

then it is straightforward to verify that $\mathcal{B}(\mathbb{S}, \mathbb{S}')$ is nothing but the unfolding game $\mathcal{U}^\nu(F)$. $\lhd$

The following proposition substantiates the slogan that 'ν means unfolding, μ means finite unfolding'.

Theorem 3.14 *Let S be some set and let $F : \wp(S) \to \wp(S)$ be a monotone operation. Then*

1. $\mathrm{GFP}.F = \{s \in S \mid s \in \mathrm{Win}_\exists(\mathcal{U}^\nu(F))\}$,
2. $\mathrm{LFP}.F = \{s \in S \mid s \in \mathrm{Win}_\exists(\mathcal{U}^\mu(F))\}$,

Proof. For the inclusion $\supseteq$ of part 1, it suffices to prove that $W := S \cap \mathrm{Win}_\exists(\mathcal{U}^\nu(F))$ is a postfixpoint of F:

$$W \subseteq F(W). \tag{18}$$

Let s be an arbitrary point in W, and suppose that $\exists$'s winning strategy tells her to choose $A \subseteq S$ at position s. Then no matter what element $s_1 \in A$ is picked by $\forall$, $\exists$ can continue the match and win. Hence, all elements of A are winning positions for $\exists$. But from $A \subseteq W$ it follows that $F(A) \subseteq F(W)$, and by the legitimacy of $\exists$'s move A at s it holds that $s \in F(A)$. We conclude that $s \in F(W)$, which proves (18).

For the converse inclusion $\subseteq$ of part 1 of the proposition, take an arbitrary point $s \in \mathrm{GFP}.F$. We need to provide $\exists$ with a winning strategy in the unfolding game $\mathcal{U}^\nu(F)$ starting at s. This strategy is actually as simple as can be: $\exists$ should always play $\mathrm{GFP}.F$. Since $\mathrm{GFP}.F = F(\mathrm{GFP}.F)$, this strategy prescribes legitimate moves for $\exists$ at every point in $\mathrm{GFP}.F$. And, if she sticks to this strategy, $\exists$ will stay alive forever and thus win the match, no matter what $\forall$'s responses are.

For the second part of the theorem, let W denote the set $\mathrm{Win}_\exists(\mathcal{U}^\mu(F))$ of $\exists$'s winning positions in $\mathcal{U}^\mu(F)$. We first prove the inclusion $W \subseteq \mathrm{LFP}.F$. Clearly it suffices to show that all points outside the set $\mathrm{LFP}.F$ are winning positions for $\forall$.

Consider a point $s \notin \mathrm{LFP}.F$. If $s \notin F(A)$ for any $A \subseteq S$ then $\exists$ is stuck, hence loses immediately, and we are done. Otherwise, suppose that $\exists$ starts a match of $\mathcal{U}^\mu(F)$ by playing some set $B \subseteq S$ with $s \in F(B)$. We claim that B is not a subset of $\mathrm{LFP}.F$, since otherwise we would have $F(B) \subseteq F(\mathrm{LFP}.F) \subseteq \mathrm{LFP}.F$; which would contradict the fact that $s \notin \mathrm{LFP}.F$. But if $B \not\subseteq \mathrm{LFP}.F$ then $\forall$ may continue the match by choosing a point $s_1 \in B \setminus \mathrm{LFP}.F$. Now $\forall$ can use the same strategy from s_1 as he used from s, and so on. This strategy guarantees that either $\exists$ gets stuck after finitely many rounds (in case $\forall$ manages to pick an s_n for which there is no A such that $s_n \in F(A_n)$), or else the match will last forever. In both cases $\forall$ wins the match.

The other inclusion $\subseteq$ of part 2 is easily proved using the ordinal approximation of least fixpoints. Using the fact that $\mathrm{LFP}.F = \bigcup\{F^\alpha_\mu(\varnothing) \mid \alpha \text{ an ordinal }\}$, it suffices to prove that

$$F^\alpha_\mu(\varnothing) \subseteq \mathrm{Win}_\exists(\mathcal{U}^\mu(F))$$

for all α. This proof proceeds by a transfinite induction, of which we only provide the case for successor ordinals. Let $\alpha = \beta + 1$ be some successor ordinal and inductively assume that $\exists$ has a winning strategy f_t for every point $t \in F^\beta_\mu(\varnothing)$. We need to provide her with a strategy which is winning from an arbitrary position $s \in F^\alpha_\mu(\varnothing)$. By definition $F^\alpha_\mu(\varnothing) = F(F^\beta_\mu(\varnothing))$, so $\exists$ may legitimately choose the set $F^\beta_\mu(\varnothing)$ as her first move at position s, and then, confronted with $\forall$ choosing a point, say, t, from $F^{\beta}_{\mu}(\varnothing)$, continue with the strategy f_t. It is almost immediate that this is a winning strategy for $\exists$. QED

Remark 3.15 Note that the proof of Theorem 3.14 witnesses a fundamental *asymmetry* in the treatment of least and greatest fixpoints in the unfolding game. In order to show that a state s belongs to one of the extremal fixpoints of a monotone map F, in both cases the approach is 'from below', i.e., in the game $\exists$ tries to provide positive evidence that s belongs to the given kind of fixpoint. However, in the case of the *least* fixpoint, this evidence from below consists of the ordinal approximations of LFP.F, whereas in the case of the *greatest* fixpoint, in the end what she tries to show is that the point in question belongs to *some* postfixpoint. Phrased differently, the game characterization of the greatest fixpoint of F uses the Knaster-Tarski characterization (16), whereas the characterization of the least fixpoint uses the ordinal approximation of Corollary 3.7. $\lhd$

3.3 Vectorial 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 $f(c) := (f_1(c), \ldots, f_n(c))$. Monotonicity of f is an easy consequence of the monotonicity of the maps f_i separately, and so by completeness of $\mathbb{C}$, f has a least and a greatest fixpoint. In this context we will also use vector notation, for instance writing

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

for LFP.f. 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 $f_1, \ldots, f_n$.

The answer to this question is positive, and the basic observation facilitating the computation of multi-dimensional fixpoints is the following so-called *Bekič principle*.

Proposition 3.16 *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. Define $\mathbb{D} := \mathbb{D}_1 \times \mathbb{D}_2$, and let $f : D \to D$ be given by putting $f(d) := (f_1(d), f_2(d))$. Then f is clearly monotone, and so it has both a least and a greatest fixpoint.

By the order duality principle it suffices to consider the case $\eta = \mu$ of least fixed points only. Suppose that (a_1, a_2) is the least fixpoint of f, and let b_1 and b_2 be given by

$$\begin{cases} b_1 & := & \mu x. f_1(x, \mu y. f_2(x, y)), \\ b_2 & := & \mu y. f_2(\mu x. f_1(x, y), y). \end{cases}$$

Then we need to show that $a_1 = b_1$ and $a_2 = b_2$.

By definition of (a_1, a_2) we have

$$\begin{cases} a_1 &= f_1(a_1, a_2), \\ a_2 &= f_2(a_1, a_2), \end{cases}$$

whence we obtain

$$\begin{cases} \mu x.f_1(x, a_2) &\leq a_1 \qquad \text{and} \\ \mu y.f_2(a_1, y) &\leq a_2, \end{cases}$$

From this we obtain by monotonicity that

$$f_2(\mu x.f_1(x, a_2), a_2) \leq f_2(a_1, a_2) = a_2,$$

so that we find $b_2 \leq a_2$ by definition of b_2. Likewise we may show that $b_1 \leq a_1$.

Conversely, by definition of b_1 and b_2 we have

$$\begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} f_1(b_1, \mu y.f_2(b_1, y)) \\ f_2(\mu x.f_1(x, b_2), b_2) \end{pmatrix}.$$

Then with $c_2 := \mu y.f_2(b_1, y)$, we have $b_1 = f_1(b_1, c_2)$. Also, by definition of c_2 as a fixpoint, $c_2 = f_2(b_1, c_2)$. Putting these two identities together, we find that

$$\begin{pmatrix} b_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} f_1(b_1, c_2) \\ f_2(b_1, c_2) \end{pmatrix} = f \begin{pmatrix} b_1 \\ c_2 \end{pmatrix}.$$

Hence by definition of (a_1, a_2), we find that $a_1 \leq b_1$ (and that $a_2 \leq c_2$, but that is of less interest now). Analogously, we may show that $a_2 \leq b_2$. QED

Proposition 3.16 allows us to compute the least and greatest fixpoints of any monotone map f on a finite product of complete lattices in terms of the least and greatest fixpoints of operations on the factors of the product, through a *elimination method* that is reminiscent of Gaussian elimination in linear algebra.

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

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

We may define

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

and then use Proposition 3.16, with $\mathbb{D}_1 = \mathbb{C}_1 \times \cdots \times \mathbb{C}_n$, and $\mathbb{D}_2 = \mathbb{C}_{n+1}$, to obtain

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

We may then inductively assume to have obtained the tuple $(a_1, \dots, a_n)$. Finally, we may compute $a_{n+1} := g_{n+1}(a_1, \dots, a_n)$.

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 fixpoint language, then each of the components a_i of the extremal fixpoints of f can also be expressed in this language.

3.4 Algebraic semantics for the modal μ-calculus

Basic definitions

In order to define the algebraic semantics of the modal μ-calculus, we need to consider formulas as *operations* on the power set of the (state space of a) transitions system, and we have to prove that such operations indeed have least and greatest fixpoints. In order to make this precise, we need some preliminary definitions.

Definition 3.17 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 $\varphi^{\mathbb{S}}_x : \wp(S) \to \wp(S)$ defined by

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

Example 3.18 a) Where $\varphi_a = p \lor x$ we have $(\varphi_a)^{\mathbb{S}}_x(X) = [\![p \lor x]\!]^{\mathbb{S}[x \mapsto X]} = V(p) \cup X$.

b) Where $\varphi_b = \overline{x}$ we have $(\varphi_b)^{\mathbb{S}}_x(X) = [\![\overline{x}]\!]^{\mathbb{S}[x \mapsto X]} = S \setminus X$.

c) Where $\varphi_c = p \lor \Diamond_d x$ we find $(\varphi_c)^{\mathbb{S}}_x(X) = [\![p \lor \Diamond_d x]\!]^{\mathbb{S}[x \mapsto X]} = V(p) \cup \langle R_d\rangle X$.

d) Where $\varphi_d = \Diamond_d \overline{x}$ we find $(\varphi_d)^{\mathbb{S}}_x(X) = [\![\Diamond_d \overline{x}]\!]^{\mathbb{S}[x \mapsto X]} = \langle R_d\rangle(S \setminus X)$. $\lhd$

Remark 3.19 Clearly, relative to a model $\mathbb{S}$, X is a fixpoint of $\varphi^{\mathbb{S}}_x$ iff $X = \varphi^{\mathbb{S}}_x(X)$; a prefixpoint iff $\varphi^{\mathbb{S}}_x(X) \subseteq X$ and a postfixpoint iff $X \subseteq \varphi^{\mathbb{S}}_x(X)$.

Writing $\mathbb{S} \Vdash \varphi$ for $S = [\![\varphi]\!]^{\mathbb{S}}$, an alternative but equivalent way of formulating this is to say that in $\mathbb{S}$, X is a *prefixpoint of a formula* $\varphi(x)$ iff $\mathbb{S}[x \mapsto X] \Vdash \varphi \to x$, a *postfixpoint* iff $\mathbb{S}[x \mapsto X] \Vdash x \to \varphi$, and a *fixpoint* iff $\mathbb{S}[x \mapsto X] \Vdash x \leftrightarrow \varphi$. $\lhd$

Example 3.20 Consider the formulas of Example 3.18.

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 $\overline{x}$ has no fixpoints at all. (Otherwise X would be identical to its own complement relative to some nonempty set S.)

c) Two fixpoints of φ_c were already given in Example 2.1.

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. $\lhd$

In particular, it is not the case that every formula has a least fixpoint. If we can guarantee that the induced function $\varphi_x^{\mathbb{S}}$ of φ is monotone, however, then the Knaster-Tarski theorem (Theorem 3.4) provides both least and greatest fixpoints of $\varphi_x^{\mathbb{S}}$. Precisely for this reason, in the definition of fixpoint formulas, we imposed the condition in the clauses for $\eta x.\varphi$, that x may only occur positively in φ. As we will see, this condition on x guarantees monotonicity of the function $\varphi_x^{\mathbb{S}}$.

Definition 3.21 Given a $\mu\mathtt{ML}_{\mathsf{D}}$-formula φ and a labelled transition system $\mathbb{S} = \langle S, V, R\rangle$, we define the *meaning* $[\![\varphi]\!]^{\mathbb{S}}$ of φ in $\mathbb{S}$, together with the map $\varphi_x^{\mathbb{S}} : \wp(S) \to \wp(S)$ by the following simultaneous formula induction:

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

The map $\varphi_x^{\mathbb{S}}$, for $x \in \mathsf{Prop}$, is given by $\varphi_x^{\mathbb{S}}(X) = [\![\varphi]\!]^{\mathbb{S}[x \mapsto X]}$. ◁

Theorem 3.22 *Let φ be an $\mu\mathtt{ML}_{\mathsf{D}}$-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. We leave the details of this proof to the reader (see Exercise 3.2). QED

Negation in the modal μ-calculus

It follows from the definitions that the set $\mu\mathtt{ML}_{\mathsf{D}}$ is closed under taking *negations*. Informally, let $\sim\varphi$ be the result of simultaneously replacing all occurrences of $\top$ with $\bot$, of p with $\overline{p}$ and vice versa (for *free* variables p), of $\wedge$ with $\vee$, of $\Box_d$ with $\Diamond_d$, of μx with νx, and vice versa, while leaving occurrences of bound variables unchanged. As an example, $\sim(\mu x.p \vee \Diamond x) = \nu x.\overline{p} \wedge \Box x$. Formally, it is easiest to define $\sim\varphi$ via the *Boolean dual* of φ.

Definition 3.23 Given a modal fixpoint formula φ, we define its *Boolean dual* φ^∂ inductively as follows:

$$
\begin{array}{lcl@{\qquad\qquad}lcl}
\bot^\partial & := & \top & \top^\partial & := & \bot \\
(\overline{p})^\partial & := & \overline{p} & p^\partial & := & p \\
(\varphi \vee \psi)^\partial & := & \varphi^\partial \wedge \psi^\partial & (\varphi \wedge \psi)^\partial & := & \varphi^\partial \vee \psi^\partial \\
(\Box_d \varphi)^\partial & := & \Diamond_d \varphi^\partial & (\Diamond_d \varphi)^\partial & := & \Box_d \varphi^\partial \\
(\mu x.\varphi)^\partial & := & \nu x.\varphi^\partial & (\nu x.\varphi)^\partial & := & \mu x.\varphi^\partial
\end{array}
$$

Based on this definition, we define the formula $\sim\varphi$ as the formula $\varphi^\partial[p \rightleftharpoons \overline{p} \mid p \in FV(\varphi)]$ that we obtain from φ^∂ by replacing all occurrences of p with $\overline{p}$, and vice versa, for all free proposition letters $p \in FV(\varphi)$. $\lhd$

The following proposition states that $\sim$ functions as a standard Boolean negation. We let $\sim_S X := S \setminus X$ denote the complement of X in S.

Proposition 3.24 *Let φ be a modal fixpoint formula. Then $\sim\varphi$ corresponds to the negation of φ, that is,*

$$[\![\sim\varphi]\!]^{\mathbb{S}} = \sim_S [\![\varphi]\!]^{\mathbb{S}} \tag{19}$$

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

Proof. We first show, by induction on φ, that φ^∂ corresponds to the Boolean dual of φ. For this purpose, given a labelled transition system $\mathbb{S} = (S, R, V)$, we let $\mathbb{S}^\sim$ denote the *complemented* model, that is, the structure $(S, R, V^\sim)$, where $V^\sim(p) := \sim_S V(p)$. Then we claim that

$$[\![\varphi^\partial]\!]^{\mathbb{S}} = \sim_S [\![\varphi]\!]^{\mathbb{S}^\sim}, \tag{20}$$

and we prove this statement by induction on the complexity of φ. Leaving all other cases as exercises for the reader, we concentrate on the inductive case where φ is of the form $\mu x.\psi$. Then we may show that, for an arbitrary subset $U \subseteq S$:

$$(\psi^\partial)^{\mathbb{S}}_x(U) = [\![\psi^\partial]\!]^{\mathbb{S}[x\mapsto U]} = \sim_S [\![\psi]\!]^{(\mathbb{S}[x\mapsto U])^\sim} = \sim_S [\![\psi]\!]^{(\mathbb{S}^\sim[x\mapsto \sim_S U])} = (\psi^{\mathbb{S}^\sim}_x)^\partial(U),$$

where we use the inductive hypothesis on ψ and $\mathbb{S}[x \mapsto U]$ in the second equality. Clearly this implies that

$$(\psi^\partial)^{\mathbb{S}}_x = (\psi^{\mathbb{S}^\sim}_x)^\partial. \tag{21}$$

We now turn to the proof of (20) for the case where $\varphi = \mu x.\psi$:

$$\begin{aligned}
[\![(\mu x.\psi)^\partial]\!]^{\mathbb{S}} &= [\![\nu x.\psi^\partial]\!]^{\mathbb{S}} && \text{(Definition } (\mu x.\psi)^\partial\text{)} \\
&= \mathrm{GFP}.(\psi^\partial)^{\mathbb{S}}_x && \text{(Theorem 3.22)} \\
&= \mathrm{GFP}.(\psi^{\mathbb{S}^\sim}_x)^\partial && \text{(Equation (21))} \\
&= \sim_S \mathrm{LFP}.\psi^{\mathbb{S}^\sim}_x && \text{(Proposition 3.11)} \\
&= \sim_S [\![\mu x.\psi]\!]^{\mathbb{S}^\sim} && \text{(Theorem 3.22)}
\end{aligned}$$

To obtain (19) from (20), first observe that we have

$$[\![\chi[p \rightleftharpoons \overline{p} \mid p \in FV(\chi)]]\!]^{\mathbb{S}} = [\![\chi]\!]^{\mathbb{S}^\sim} \tag{22}$$

for any formula χ. But then, taking φ^∂ for χ, we find that

$$[\![\sim\varphi]\!]^{\mathbb{S}} = [\![\varphi^\partial]\!]^{\mathbb{S}^\sim} = \sim_S [\![\varphi]\!]^{(\mathbb{S}^\sim)^\sim} = \sim_S [\![\varphi]\!]^{\mathbb{S}},$$

where the first equality holds by (22) and the definition of $\sim\varphi$, the second equality is (20), and the third equality follows from the trivial observation that $(\mathbb{S}^\sim)^\sim = \mathbb{S}$. QED

Remark 3.25 It follows from the Proposition above that we could indeed have based the language of the modal μ-calculus on a smaller alphabet of primitive symbols. Given a set D of atomic actions, we could have defined the set of modal fixpoint formulas using the following induction:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \Diamond_d \varphi \mid \mu x.\varphi$$

where p and x are propositional variables, $d \in \mathsf{D}$, 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 $\Box_d$ are defined using the standard abbreviations, while for the greatest fixpoint operator we may put

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

Note the *triple* use of the negation symbol here, which can be explained by Proposition 3.11 and the observation that we may think of $\neg\varphi(\overline{x})$ as the formulas φ^∂. $\lhd$

Other immediate consequences

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

Proposition 3.26 *Every fixpoint formula is equivalent to a clean one.*

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

Proposition 3.27 *Every fixpoint formula is equivalent to a guarded one.*

Proof.(Sketch) We prove this proposition by formula induction. Clearly the only nontrivial 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 where $\eta = \mu$, leaving the very similar case where $\eta = \nu$ as an exercise. Clearly, using the laws of classical propositional logic, we may bring the formula $\overline{\delta}$ into conjunctive normal form

$$(x \vee \alpha_1(x)) \wedge \cdots \wedge (x \vee \alpha_n(x)) \wedge \beta(x), \tag{23}$$

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 (23) 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

$$\mu x.\delta'(x) \;\equiv\; \mu x.\alpha(x) \wedge \beta(x). \tag{24}$$

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 φ is in fact 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 3.28 *Every fixpoint formula is equivalent to a clean, guarded one.*

Remark 3.29 The equivalences of the above propositions are in fact *effective* in the sense that there are algorithms for computing an equivalent clean and/or guarded equivalent to an arbitrary formula in $\mu\mathtt{ML}$. It is an interesting question what the complexity of these algorithms is, and what the minimum *size* of the equivalent formulas is. We will return to this issue later on, but already mention here that there are formulas that are exponentially smaller than any of their clean equivalents. The analogous question for guarded transformations, i.e., constructions that provide guarded equivalents to an arbitrary formula, is open. $\lhd$

3.5 Adequacy

In this section we prove the *equivalence* of the two semantic approaches towards the modal μ-calculus. Since the algebraic semantics is usually taken to be the more fundamental notion, we refer to this result as the *Adequacy Theorem* stating, informally, that games are an adequate way of working with the algebraic semantics.

Theorem 3.30 (Adequacy) *Let ξ be a clean $\mu\mathtt{ML}_{\mathsf{D}}$-formula. Then for all labelled transition systems $\mathbb{S}$ and all states s in $\mathbb{S}$:*

$$s \in [\![\xi]\!]^{\mathbb{S}} \iff (\xi, s) \in \mathrm{Win}_{\exists}(\mathcal{E}(\xi, \mathbb{S})). \tag{25}$$

Proof. The theorem is proved by induction on the complexity of ξ. We only discuss the inductive steps where ξ is of the form $\eta x.\delta$ (with η denoting either μ or ν), leaving the other cases as exercises to the reader. Our proof for these inductive cases will involve *three* games: the unfolding game for $\delta_x^{\mathbb{S}}$, and the evaluation games for ξ and δ, respectively. It is based on the following two key observations, concerning, respectively, the nature of the unfolding game for $\delta_x^{\mathbb{S}}$, and its role in the semantics for $\eta x.\delta$, and the similarity between the evaluation games for ξ and for δ.

Starting with the first observation, note that by definition of the algebraic semantics of the fixpoint operators, the set $[\![\eta x.\delta]\!]^{\mathbb{S}}$ is the least/greatest fixed point of the map $\delta^{\mathbb{S}}_x : \wp(S) \to \wp(S)$, and that by our earlier Theorem 3.14 on unfolding games, we have

$$[\![\eta x.\delta]\!]^{\mathbb{S}} = \mathrm{Win}_\exists(\mathcal{U}^\eta(\delta^{\mathbb{S}}_x)) \cap S. \tag{26}$$

Hence, in order to prove (25), it suffices to show that, for any state s_0:

$$s_0 \in \mathrm{Win}_\exists(\mathcal{U}^\eta(\delta^{\mathbb{S}}_x)) \iff (\xi, s_0) \in \mathrm{Win}_\exists(\mathcal{E}(\xi, \mathbb{S})). \tag{27}$$

In other words, the crucial tasks in the proof of this inductive step concern the transformation of a winning strategy for $\exists$ in the unfolding game $\mathcal{U}^\eta(\delta^{\mathbb{S}}_x)@s_0$ to a winning strategy for her in the evaluation game $\mathcal{E}(\xi, \mathbb{S})@(\xi, s_0)$, and vice versa.

Given the importance of the unfolding game for $\delta^{\mathbb{S}}_x$ then, let us look at it in a bit more detail. Note that a round of this game, starting at position $s \in S$, consists of $\exists$ picking a subset $A \subseteq S$ that is subject to the constraint that $s \in \delta^{\mathbb{S}}_x(A) = [\![\delta]\!]^{\mathbb{S}[x \mapsto A]}$. But here the inductive hypothesis comes into play: it implies that, for all $A \subseteq S$, we have

$$s \in \delta^{\mathbb{S}}_x(A) \iff (\delta, s) \in \mathrm{Win}_\exists(\mathcal{E}(\delta, \mathbb{S}[x \mapsto A])). \tag{28}$$

In other words, each round of the unfolding game for the map $\delta^{\mathbb{S}}_x$ crucially involves the evaluation game for the formula δ, played on some x-variant $\mathbb{S}[x \mapsto A]$ of $\mathbb{S}$.

This leads us to the comparison between the games $\mathcal{G} := \mathcal{E}(\xi, \mathbb{S})$ and $\mathcal{G}_A := \mathcal{E}(\delta, \mathbb{S}[x \mapsto A])$. The second key observation in the inductive step for the fixpoint operators is that these games are *very* similar indeed. For a start, the positions of the two games are essentially the same. Positions of the form (ξ, t), which exist in the first game but not in the second, are the only exception — but in $\mathcal{G}$, any position (ξ, t) is immediately and automatically succeeded by the position (δ, t) which does exist in the second game. What is important is that the positions for $\exists$ are exactly the same in the two games, and thus we may apply her positional strategies for the one game in the other game as well. The only real difference between the games shows up in the rule concerning positions of the form (x, u). In $\mathcal{G}_A$, x is a *free* variable ($x \in FV(\delta)$), so in a position (x, u) the game is over, the winner being determined by u being a member of A or not. In $\mathcal{G}$ however, x is *bound*, so in position (x, u), the variable x will get unfolded to δ.

Combining these two observations, the key insight in the proof of (27) will be to think of $\mathcal{E}(\xi, \mathbb{S})$ as a variant of the unfolding game $\mathcal{U} := \mathcal{U}^\eta(\delta^{\mathbb{S}}_x)$ where each round of $\mathcal{U}$ corresponds to a version of the game $\mathcal{G}_T$, with T being the subset of S picked by $\exists$ in $\mathcal{U}$. We are now ready for the details of the proof of (27).

For the direction from left to right of (27), suppose that $\exists$ has a winning strategy in the game $\mathcal{U}$ starting at some position s_0. Without loss of generality (see Exercise 3.6) we may assume that this strategy is *positional*. Thus we may represent it as a map $T : S \to \wp(S)$, where we will write T_s rather than $T(s)$. By the legitimacy of this strategy, for every $s \in \mathrm{Win}_\exists(\mathcal{U})$ it holds that $s \in \delta^{\mathbb{S}}_x(T_s)$. So by the inductive hypothesis (28), for each such s we may assume the existence of a winning strategy f_s for $\exists$ in the game $\mathcal{G}_{T_s}@(\delta, s)$. Given the similarities between the games $\mathcal{G}$ and $\mathcal{G}_{T_s}$ (see the discussion above), this strategy is also applicable in the game $\mathcal{G}@(\delta, s)$, at least, until a new position of the form (x, t) is reached.

This suggests the following strategy g for $\exists$ in $\mathcal{G}@(\xi, s_0)$:

1. after the initial automatic move, the position of the match is (δ, s_0); $\exists$ first plays her strategy f_{s_0};
2. each time a position (x, s) is reached, the match automatically moves to position (δ, s), where we distinguish cases:
   (a) if $s \in \mathrm{Win}_\exists(\mathcal{U})$ then $\exists$ continues with f_s;
   (b) if $s \notin \mathrm{Win}_\exists(\mathcal{U})$ then $\exists$ continues with a random strategy.

First we show that this strategy guarantees that whenever a position of the form (x, s) is visited, s belongs to $\mathrm{Win}_\exists(\mathcal{U})$, so that case (b) mentioned above never occurs. The proof is by induction on the number of positions (x, s) that have been visited already. For the inductive step, if s is a winning position for $\exists$ in $\mathcal{U}$, then, as we saw, f_s is a winning strategy for $\exists$ in the game $\mathcal{G}_{T_s}@(\delta, s)$. This means that if a position of the form (x, t) is reached, the variable x must be *true* at t in the model $\mathbb{S}[x \mapsto T_s]$, and so t must belong to the set T_s. But by assumption of the map $T : S \to \wp(S)$ being a winning strategy in $\mathcal{U}$, any element of T_s is again a member of $\mathrm{Win}_\exists(\mathcal{U})$.

In fact we have shown that every unfolding of the variable x in $\mathcal{G}$ marks a new round in the unfolding game $\mathcal{U}$. To see why the strategy g guarantees a win for $\exists$ in $\mathcal{G}@(\xi, s_0)$, consider an arbitrary $\mathcal{G}@(\xi, s_0)$-match π in which $\exists$ plays g. Distinguish cases.

First suppose that x is unfolded only finitely often. Let (x, s) be the last basic position in π where this happens. Given the similarities between the games $\mathcal{G}$ and $\mathcal{G}_{T_s}$, the match from this moment on can be seen as both a g-guided $\mathcal{G}$-match and an f_s-guided $\mathcal{G}_{T_s}$-match. As we saw, f_s is a winning strategy for $\exists$ in the game $\mathcal{G}_{T_s}@(\delta, s)$. But since no further position of the form (x, t) is reached, and $\mathcal{G}$ and $\mathcal{G}_{T_s}$ only differ when it comes to x, this means that π is also a win for $\exists$ in $\mathcal{G}$.

If x is unfolded infinitely often during the match π, then by the fact that $\xi = \eta x.\delta$, it is the *highest* variable that is unfolded infinitely often. We have to distinguish the case where $\eta = \nu$ from that where $\eta = \mu$. In the first case, $\exists$ is the winner of the match π, and we are done. If $\eta = \mu$, however, x is a least fixpoint variable, and so $\exists$ would lose the match π. We therefore have to show that this situation cannot occur. Suppose for contradiction that $s_1, s_2, \ldots$ are the positions where x is unfolded. Then it is easy to verify that the sequence $s_0 T_{s_0} s_1 T_{s_1} \ldots$ constitutes a $\mathcal{U}$-match in which $\exists$ plays her strategy T. But this is not possible, since T was assumed to be a *winning* strategy for $\exists$ in the least fixpoint game $\mathcal{U} = \mathcal{U}^\mu(\delta^{\mathbb{S}}_x)$.

For the converse implication of (27), we will show how each of $\exists$'s positional winning strategies f in $\mathcal{G}$ induces a positional strategy for her in $\mathcal{U}$, and that this strategy U_f is winning for her starting at every position $s \in W := \{s \in S \mid (\xi, s) \in \mathrm{Win}_\exists(\mathcal{G})\}$.

So fix a positional winning strategy f for $\exists$ in $\mathcal{G}$; that is, $\exists$ is guaranteed to win any f-guided match starting at a position $(\varphi, t) \in \mathrm{Win}_\exists(\mathcal{G})$. Observe that, as discussed above, we may and will treat f as a positional strategy in each of the games $\mathcal{G}_A$ as well.

Given a state $s \in W$, we let $\mathbb{T}_f(s)$ be the *strategy tree* induced by f in $\mathcal{G}_A@(\delta, s)$, where A is some arbitrary subset of S. That is, the nodes of $\mathbb{T}_f$ consist of all f-guided finite matches in $\mathcal{G}_A$ that start at (δ, s). In more detail, the root of this tree is the single-position match (δ, s); to define the successor relation of $\mathbb{T}_f$, let Σ be an arbitrary f-guided match starting at position *first*$(\Sigma) = (\delta, s)$. If *last*(Σ) is a position owned by $\exists$, then Σ will have a single successor in $\mathbb{T}_f$, viz., the unique extension of Σ with the position $f(\Sigma)$ picked by f. On the other hand, if $last(\Sigma)$ is owned by $\forall$, then any possible continuation $\Sigma \cdot b$, where b is an admissible position picked by $\forall$, is a successor of Σ.

We let $U_f(s)$ be the set of states u such that the position (x, u) occurs as the last element $(x, u) = last(\Sigma)$ of some match Σ in $\mathbb{T}_f(s)$. It is easy to see that any $\mathcal{G}_A$-match Σ ending in a position of the form (x, u), is finished immediately, and thus provides a *leaf* of the tree $\mathbb{T}_f$. It is also an easy consequence of the definitions that, whenever $t \in U_f(s)$ for some $s \in W$, then there is an f-guided match $\Sigma_{s,t}$ such that $first(\Sigma_{s,t}) = (\delta, s)$ and $last(\Sigma_{s,t}) = (x, t)$. Note that this match $\Sigma_{s,t}$ can be seen both as a (full) $\mathcal{G}_A$-match and as a (partial) $\mathcal{G}$-match.

Given our definition of a set $U_f(s) \subseteq S$ for every $s \in W$, in effect we have defined a map $U_f : W \to \wp(S)$. Viewing this map U_f as a positional strategy for $\exists$ in $\mathcal{U}$, we claim that in fact it is a *winning* strategy for her in $\mathcal{U}@s_0$. Before proving this, we state and prove two auxiliary claims on U_f. First we observe that

$$\text{if } s \in W \text{ then } s \in \delta_x^{\mathbb{S}}(U_f(s)). \tag{29}$$

For a proof of (29), it is obvious from the definition of $U_f(s)$ that f is a positional winning strategy for $\exists$ in $\mathcal{G}_{U_f(s)} = \mathcal{E}(\delta, \mathbb{S}[x \mapsto U_f(s)])$ starting at (δ, s). But then by the inductive hypothesis on δ we obtain that $\mathbb{S}[x \mapsto U_f(s)], s \Vdash \delta$, or, equivalently, $s \in \delta_x^{\mathbb{S}}(U_f(s))$.

Second, we claim that

$$\text{if } s \in W \text{ then } U_f(s) \subseteq W. \tag{30}$$

To see this, first note that if $s \in W$ then by definition $(\xi, s) \in \mathrm{Win}_\exists(\mathcal{G})$; but from this it is immediate that $(\delta, s) \in \mathrm{Win}_\exists(\mathcal{G})$, and since we assumed f to be a positional winning strategy for $\exists$ in $\mathcal{G}$, it follows by definition of $U_f(s)$ that for every $u \in U_f(s)$ the position (x, u) is winning for $\exists$ in $\mathrm{Win}_\exists(\mathcal{G})$. But from this it is easy to derive that both (δ, u) and (ξ, u) are winning position for $\exists$ in $\mathcal{G}$ as well. The latter fact then shows that $u \in W$ and since u was an arbitrary element of $U_f(s)$, (30) follows.

We can now prove that U_f is a winning strategy for $\exists$ in $\mathcal{U}@s_0$. First of all, it follows from (29) that $U_f(s)$ is a legitimate move in $\mathcal{U}$ for every position $s \in W$. From this and (30) we may conclude that $\exists$ never gets stuck in an U_f-guided $\mathcal{U}$-match starting at s_0; that is, she wins every *finite* U_f-guided $\mathcal{U}$-match. In case $\eta = \nu$ this suffices, since in $UG^{\nu}(\delta_x^{\mathbb{S}})$ all infinite matches are won by $\exists$.

Where $\eta = \mu$ we have a bit more work to do, since in this case all infinite matches of $\mathcal{U}^{\mu}(\delta_x^{\mathbb{S}})$ are won by $\forall$. Suppose for contradiction that $\Sigma = s_0 U_f(s_0) s_1 U_f(s_1) \cdots$ would be an infinite U_f-guided match of $\mathcal{U}^{\mu}(\delta_x^{\mathbb{S}})$. Then for every $i \in \omega$ we have that $s_{i+1} \in U_f(s_i)$, so that there is a partial f-guided match $\Sigma_i = \Sigma_{s_i s_{i+1}}$ with $first(\Sigma_i) = (\delta, s_i)$ and $last(\Sigma_i) = (x, s_{i+1})$. But then it is straightforward to verify that the infinite match $\Sigma_{\mathcal{G}} := \Sigma_0 \cdot \Sigma_1 \cdot \Sigma_2 \cdots$ we obtain by concatenating the individual f-guided matches Σ_i, constitutes an infinite f-guided $\mathcal{G}$-match with $first(\Sigma_{\mathcal{G}}) = first(\Sigma_0) = (\xi, s_0)$. Since the highest fixpoint variable unfolded infinitely often during $\Sigma_{\mathcal{G}}$ obviously would be x, this match would be lost by $\exists$. Here we arrive at the desired contradiction, since $(\xi, s_0) \in \mathrm{Win}_\exists(\mathcal{G})$, and f was assumed to be a positional winning strategy in $\mathcal{G}$. QED

Convention 3.31 In the sequel we will use the Adequacy Theorem without further notice. Also, we will write $\mathbb{S}, s \Vdash \varphi$ in case $s \in [\![\varphi]\!]^{\mathbb{S}}$, or, equivalentlu, $\mathbb{S}, s \Vdash_g \varphi$.

Notes

What we now call the Knaster-Tarski Theorem (Theorem 3.4) was first proved by Knaster [14] in the context of power set algebras, and subsequently generalized by Tarski [27] to the setting of complete lattices. The Bekič principle (Proposition 3.16) stems from an unpublished technical report.

▶ `more notes and references to be supplied`

As far as we know, the results in section 3.2 on the duality between the least and the greatest fixpoint of a monotone map on a complete Boolean algebra, are folklore. The characterization of least and greatest fixpoints in game-theoretic terms is fairly standard in the theory of (co-)inductive definitions, see for instance Aczel [1]. The equivalence of the algebraic and the game-theoretic semantics of the modal μ-calculus (here formulated as the Adequacy Theorem 3.30) was first established by Emerson & Jutla [11].

Exercises

Exercise 3.1 Prove Proposition 3.6: show that monotone maps on complete lattices are inductive.

Exercise 3.2 Prove Theorem 3.22.
(Hint: given complete lattices $\mathbb{C}$ and $\mathbb{D}$, and a monotone map $f : C \times D \to C$, show that the map $g : D \to C$ given by

$$g(d) := \mu x.f(x, d)$$

is monotone. Here $\mu x.f(x, d)$ is the least fixpoint of the map $f_d : C \to C$ given by $f_d(c) = f(c, d)$.)

Exercise 3.3 Let $F : \wp(S) \to \wp(S)$ be some monotone map. A collection $\mathcal{D} \in \wp\wp(S)$ of subsets of S is *directed* if for every two sets $D_0, D_1 \in \mathcal{D}$, there is a set $D \in \mathcal{D}$ with $D_i \subseteq D$ for $i = 0, 1$. Call F *(Scott) continuous* if it preserves directed unions, that is, if $F(\bigcup \mathcal{D}) = \bigcup_{D \in \mathcal{D}} F(D)$ for every directed $\mathcal{D}$.
Prove the following:

(a) F is Scott continuous iff for all $X \subseteq S$: $F(X) = \bigcup\{F(Y) \mid Y \subseteq_\omega X\}$.
(Here $Y \subseteq_\omega X$ means that Y is a finite subset of X.)

(b) If F is Scott continuous then the unfolding ordinal of F is at most ω.

(c) Give an example of a Kripke frame $\mathbb{S} = \langle S, R\rangle$ such that the operation $[R]$ is not continuous.

(d) Give an example of a Kripke frame $\mathbb{S} = \langle S, R\rangle$ such that the operation $[R]$ has closing/unfolding ordinal $\omega + 1$.

Exercise 3.4 By a mutual induction we define, for every finite set P of propositional variables, the fragment $\mu\mathtt{ML}^C_{\mathsf{P}}$ by the following grammar:

$$\varphi ::= p \mid \psi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \Diamond\varphi \mid \mu q.\varphi',$$

where $p \in \mathsf{P}$, $\psi \in \mu\mathtt{ML}$ is a P-free formula, and $\varphi' \in \mu\mathtt{ML}^C_{\mathsf{P}\cup\{q\}}$.

Prove that for every Kripke model $\mathbb{S}$, every formula $\varphi \in \mu\mathtt{ML}^C_{\mathsf{P}}$, and every proposition letter $p \in \mathsf{P}$, the map $\varphi^{\mathbb{S}}_p : \wp(S) \to \wp(S)$ is continuous.

Exercise 3.5 Let $F : \wp(S) \to \wp(S)$ be a monotone operation, and let γ_F be its unfolding ordinal. Sharpen Corollary 3.7 by proving that the cardinality of γ_F is bounded by $|S|$ (rather than by $|\wp(S)|$).

Exercise 3.6 Prove that the unfolding game of Definition 3.12 satisfies *positional determinacy.* That is, let $\mathcal{U}^\mu(F)$ be the least fixpoint unfolding game for some monotone map $F : \wp(S) \to \wp(S)$. Prove the existence of two *positional* strategies $f_\exists : S \to \wp(S)$ and $f_\forall : \wp(S) \to S$ such that for every position p of the game, either $f_\exists$ is a winning strategy for $\exists$ in $\mathcal{U}^\mu(F)@p$, or else $f_\forall$ is a winning strategy for $\forall$ in $\mathcal{U}^\mu(F)@p$.

Exercise 3.7 Let $\mathbb{C}$ be a complete boolean algebra and let $f : C \to C$ be a monotone map. Pick an element $d \in C$ and let $\mu x.f(x)$ be the least fixpoint of f.

(a) Show that $d \wedge \mu x.f(x) = \bot$ iff $d \wedge \mu x.f(x \wedge \neg d) = \bot$, where $\mu x.f(x \wedge \neg d)$ denotes the smallest fixpoint of the map sending any element $x \in C$ to $f(x \wedge \neg d)$.

(b) Conclude that, for any formula of the form $\mu x.\varphi$ and an arbitrary formula γ: the formula $\gamma \wedge \mu x.\varphi$ is satisfiable iff the formula $\gamma \wedge \mu x.\varphi[x \wedge \neg\gamma/x]$ is satisfiable. (A formula φ is called satisfiable if there exists a pointed Kripke model such that $\mathbb{S}, s \Vdash \varphi$.)

▶ `add exercise on the closure ordinal of a formula`

▶ `add exercise on (complete) additivity`

4 Stream automata and logics for linear time

As we already mentioned in the introduction in the theory of the modal μ-calculus and other fixpoint logics a fundamental role is played by automata. As we will see further on, these devices provide a very natural generalization to the notion of a formula. This chapter gives an introduction to the theory of automata operating on (potentially infinite) objects. Whereas in the next chapters we will meet various kinds of automata for classifying trees and general transition systems, here we confine our attention to the devices that operate on *streams* or infinite words, these being the simplest nontrivial examples of infinite behavior.

Convention 4.1 Throughout this chapter (and the next), we will be dealing with some finite *alphabet* C. Generic elements of C may be denoted as $c, d, c_0, c_1, \ldots$, but often it will be convenient to think of C as a set of *colors*. In this case we will denote the elements of C with lower case roman letters that are mnemonic of the most familiar corresponding color ('b' for *blue*, 'g' for *green*, etcetera).

Definition 4.2 Given an alphabet C, a *C-stream* is just an infinite C-sequence, that is, a map $\gamma : \omega \to C$ from the natural numbers to C (see Appendix A). C-streams will also be called *infinite words* or *ω-words* over C. Sets of C-streams are called *stream languages* or *ω-languages* over C. $\lhd$

Remark 4.3 This definition is consistent with the terminology we introduced in Chapter 1. There we defined a $\wp(\mathsf{P})$-*stream* or *stream model for* P to be a Kripke model of the form $\mathbb{S} = \langle \omega, V, \mathit{Succ} \rangle$, where *Succ* is the standard successor relation on the set ω of natural numbers, and $V : \mathsf{P} \to \wp(\omega)$ is a valuation. If we represent V coalgebraically as a map $\sigma_V : \omega \to \wp(\mathsf{P})$ (cf. Remark 1.3), then in the terminology of Definition 4.2, $\mathbb{S}$ is indeed a $\wp(\mathsf{P})$-*stream*. $\lhd$

4.1 Deterministic stream automata

We start with the most general definition of a deterministic stream automaton.

Definition 4.4 Given an *alphabet* C, a *deterministic C-automaton* is a quadruple $\mathbb{A} = \langle A, \delta, \mathit{Acc}, a_I \rangle$, 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*, and $\mathit{Acc} \subseteq A^\omega$ its *acceptance condition*. The pair $\langle A, \delta \rangle$ is called the *transition diagram* of $\mathbb{A}$.

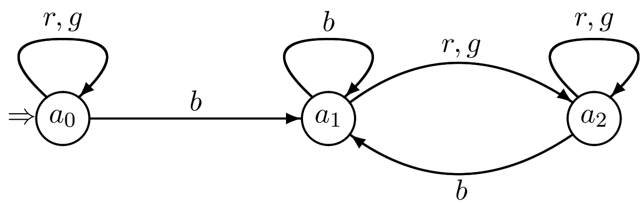
Given a finite automaton $\mathbb{A} = \langle A, \delta, \mathit{Acc}, a_I \rangle$, we may extend the map $\delta : A \times C \to A$ to a map $\widehat{\delta} : A \times C^* \to A$ by putting

$$\begin{array}{lll} \widehat{\delta}(a, \epsilon) & := & a \\ \widehat{\delta}(a, uc) & := & \delta(\widehat{\delta}(a, u), c). \end{array}$$

We will write $a\mathtt{trc}a'$ if $a' = \delta(a, c)$, and $a \overset{w}{\twoheadrightarrow} a'$ if $a' = \widehat{\delta}(a, w)$. In words, $a \overset{w}{\twoheadrightarrow} a'$ if there is a w-labelled path from a to a'.

$\lhd$

Example 4.5 The transition diagram and initial state of a deterministic automaton can nicely be represented graphically, as in the picture below, where $C = \{b, r, g\}$:



$\lhd$

An automaton comes to life 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.

Example 4.6 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 $\alpha = brgbrgbrgbrgbrgb\cdots$. Then we find that $\mathbb{A}_0$ will make an infinite series of transitions, determined by α:

$$a_0 \mathtt{tr} b a_1 \mathtt{tr} r a_2 \mathtt{tr} g a_2 \mathtt{tr} b a_1 \cdots$$

Thus the machine passes through an infinite sequence of states:

$$\rho = a_0 a_1 a_2 a_2 a_1 a_2 a_2 a_1 a_2 a_2 \ldots$$

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 $\alpha' = brbgbrgrgrgrgrgr\cdots$ the run of the automaton $\mathbb{A}_0$ looks as follows:

$$a_0 \mathtt{tr} b a_1 \mathtt{tr} r a_2 \mathtt{tr} b a_1 \mathtt{tr} g a_2 \mathtt{tr} b a_1 \mathtt{tr} r a_2 \mathtt{tr} g a_2 \mathtt{tr} r a_2 \mathtt{tr} g \cdots$$

we see that from the sixth step onwards, the machine device remains circling in its state a_2: $\cdots\ a_2 \mathtt{tr} r a_2 \mathtt{tr} g a_2 \mathtt{tr} r \cdots$. $\lhd$

Definition 4.7 The *run* of a finite automaton $\mathbb{A} = \langle A, \delta, Acc, a_I \rangle$ on a C-stream $\gamma = c_0 c_1 c_2 \ldots$ is the infinite A-sequence

$$\rho = a_0 a_1 a_2 \ldots$$

such that $a_0 = a_I$ and $a_i \mathtt{tr} c_i a_{i+1}$ for every $i \in \omega$. $\lhd$

Generally, whether or not an automaton *accepts* an infinite word, depends on the existence of a successful run — note that in the present deterministic setting, this run is unique. In order to determine which runs are successful, we need the acceptance condition.

Definition 4.8 A run $\rho \in A^\omega$ of an automaton $\mathbb{A} = \langle A, \delta, Acc, a_I \rangle$ 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 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. $\lhd$

A natural requirement on the acceptance condition is that it only depends on a bounded amount of information about the run.

Remark 4.9 In the case of automata running on *finite words*, there is a very simple and natural acceptance criterion. The point is that runs on finite words are themselves finite too. For instance, suppose that in Example 4.6 we consider the run on the finite word $brgb$:

$$a_0\mathtt{tr}ba_1\mathtt{tr}ra_2\mathtt{tr}ga_2\mathtt{tr}ba_1.$$

Then this runs *ends* in the state a_1. In this context, a natural criterion for the acceptance of the word $abca$ by the automaton is to make it dependent on the membership of this final state a_1 in a designated set $F \subseteq A$ of *accepting* states.

A structure of the form $\mathbb{A} = \langle A, \delta, F, a_I \rangle$ with $F \subseteq A$ may be called a *finite word automaton*, and we say that such a structure *accepts* a finite word w if the unique state a such that $a_I \stackrel{w}{\twoheadrightarrow} a$ belongs to F. The *language* $L(\mathbb{A})$ is defined as the set of all finite words accepted by $\mathbb{A}$. $\lhd$

4.2 Acceptance conditions

For runs on infinite words, a natural acceptance criterion would involve the collection of states that occur infinitely often in the run.

Definition 4.10 Let $\alpha : \omega \to A$ be a stream over some finite set A. Given an element $a \in A$, we define the *frequency* of a in α as $\#_a(\alpha) := |\{n \in \omega \mid \alpha(n) = a\}$—. Based on this, we set $Occ(\alpha) := \{a \in A \mid \#_a(\alpha) > 0\}$ and $Inf(\alpha) := \{a \in A \mid \#_a(\alpha) = \omega\}$ $\lhd$

In words, $Occ(\alpha)$ and $Inf(\alpha)$ denote the set of elements of A that occur in α at least once and infinitely often, respectively.

Definition 4.11 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

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

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 $\mathit{Inf}(\rho)$ 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 passes through some state in F infinitely often. This makes Büchi automata the natural analog of the automata that operate on *finite* words, see Remark 4.9.

The parity condition may be slightly more difficult to understand. The idea is to give each state a of $\mathbb{A}$ a weight $\Omega(a) \in \omega$. Then any infinite A-sequence $\alpha = a_0a_1a_2\dots$ induces an infinite sequence $\Omega(a_0)\Omega(a_1)\dots$ of natural numbers. Since the range of Ω is finite this means that there is a *largest* natural number N_α occurring infinitely often in this sequence, $N_\alpha := \max\{\Omega(a) \mid a \in \mathit{Inf}(\alpha)\}$. Now, a parity automaton accepts an infinite word iff the number N_ρ of the associated run ρ is *even*.

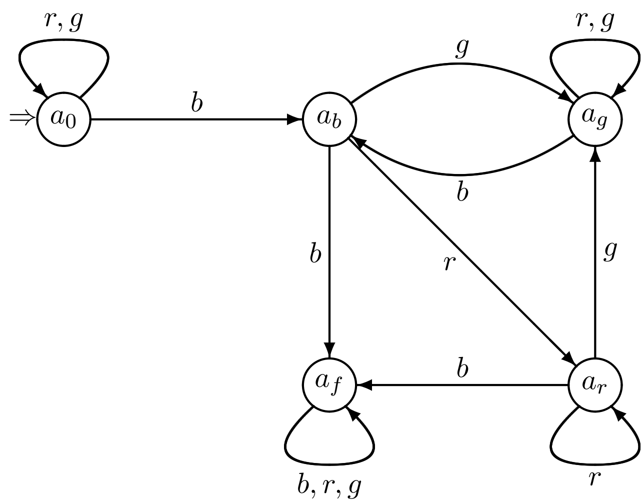
At first sight, this condition will seem rather contrived and artificial. Nevertheless, for a number of reasons the parity automaton is destined to play the leading role in these notes. Most importantly, the distinction between even and odd parities directly corresponds to that between least and greatest 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, as we will see further on.

Let us now first discuss some examples of automata with these three acceptance conditions.

Example 4.12 Suppose that we supply the device of Example 4.5 with the Büchi acceptance condition $F_0 = \{a_1\}$. That is, the resulting automaton $\mathbb{A}_0$ accepts a stream α iff the run of $\mathbb{A}_0$ passes through the state a_1 infinitely often. For instance, $\mathbb{A}_0$ will accept the word $\alpha = brgbrgbrgbrgbrgbrgb\cdots$, because the run of $\mathbb{A}_0$ is the stream $a_0a_1a_2a_2a_1a_2a_2a_1a_2a_2\dots$ which indeed contains a_1 infinitely many times. On the other hand, as we saw already, the run of $\mathbb{A}_0$ on the stream $\alpha' = brbgbrgrgrgrgrgr\cdots$ loops in state a_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 b's. ◁

Example 4.13 Consider the automaton $\mathbb{A}_1$ given by the following diagram and initial state:



As an example of a Muller acceptance condition, consider the set

$$\{\ \{a_0\}\ ,\ \{a_g\}\ ,\ \{a_b, a_g\}\ ,\ \{a_b, a_r, a_g\}\ \}$$

The resulting automaton accepts those infinite streams in which every b is followed by a finite number of r's, followed by a g. To see this, here is a brief description of the intuitive meaning of the states:

a_0 represents the situation where the automaton has not encountered any b's;

a_f is the 'faulty' state;

a_b is the state where the automaton has just processed a b; it now has to pass through a finite sequence of r's, eventually followed by a g;

a_r represents the situation where the automaton, after seeing a b, has processed a finite, non-empty, sequence of r's;

a_g is the state where the automaton, after passing the last b, has fulfilled its obligation to process a g.

We leave the details of the proof as an exercise to the reader. $\lhd$

Example 4.14 For an example of a parity automaton, consider the transition diagram of Example 4.5, and suppose that we endow the set $\{a_0, a_1, a_2\}$ with the priority map Ω given by $\Omega(a_i) = i$. Given the shape of the transition diagram, it then follows more or less directly from the definitions that the resulting automaton accepts an infinite word over $C = \{b, r, g\}$ iff it either stays in a_0, or visits a_2 infinitely often. From this one may derive that $L_\omega(\mathbb{A})$ consists of those C-streams containing infinitely many r's or infinitely many g's (or both). $\lhd$

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 4.15 *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:

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

Clearly 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 4.16 *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, we put

$$\mathcal{M}_\Omega := \{B \subseteq A \mid \max(\Omega[B]) \text{ is even }\},$$

and leave it for the reader to verify that this is the key observation in turning a parity acceptance condition into a Muller one. QED

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

Proposition 4.17 *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 finite sequence in A^*, say, $\alpha = a_1 \dots a_n$, we let $\widetilde{\alpha}$ denote the set $\{a_1, \dots, a_n\}$, and $\alpha[\triangledown/a]$ the sequence α with every occurrence of a being replaced with the symbol $\triangledown$.

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

$$A' = \{a_1 \dots a_k \triangledown a_{k+1} \dots a_m \mid A = \{a_1, \dots, a_m\}\}.$$

The intuition behind this definition is that a state in $\mathbb{A}'$ encodes information about the states of $\mathbb{A}$ that have been visited during the initial part of its run on some word. More specifically, the state $a_1 \dots a_k \triangledown a_{k+1} \dots a_m$ encodes that the states visited by $\mathbb{A}$ are $a_{n+1}, \dots, a_m$ (for some $n \leq m$, not necessarily $n = k$), and that of these, a_m is the state visited most recently, a_{m-1} the one before that, etc. The symbol $\triangledown$ marks the *previous* position of a_m in the list.

For a proper understanding of $\mathbb{A}'$ we need to go into more detail. First, for the initial position of $\mathbb{A}'$, fix some enumeration $d_1, \dots, d_m$ of A with $a_I = d_m$, and define

$$a'_I := d_1 \dots d_m \triangledown.$$

For the transition function, consider a state $\alpha = a_1 \dots a_k \triangledown a_{k+1} \dots a_m$ 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 $\triangledown$, and make the state $\delta(a_m, c)$ itself the rightmost element of the resulting sequence. Thus the $\triangledown$ in the new sequence marks the latest appearance of the state $\delta(a_m, c)$. Formally, we put

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

For an example, see 4.18 below.

Now consider the runs ρ and ρ' of $\mathbb{A}$ and $\mathbb{A}'$, respectively, on some C-stream γ. Recall that $\mathit{Inf}(\rho)$ denotes the set of states of $\mathbb{A}$ that are visited infinitely often during ρ. From a certain moment on, ρ will *only* pass through states in $\mathit{Inf}(\rho)$; let $\mathbb{A}$ continue its run until it has passed through each state in $\mathit{Inf}(\rho)$ at least one more time. It is not too hard to see that from that same moment on, ρ' will only pass through states of the form $a_1 \dots a_k \triangledown a_{k+1} \dots a_m$ such that the states in $\mathit{Inf}(\rho)$ form a final segment $a_{l+1} \dots a_m$ of the sequence $a_1 \dots a_m$. Also, since $\triangledown$ marks the previous position of a_m, it must occur before one of the a_i with $l + 1 \leq i < m$. In other words, we have

$$\mathit{Inf}(\rho') \subseteq \{\alpha \triangledown \beta \in A' \mid \widetilde{\beta} \subseteq \mathit{Inf}(\rho)\}.$$

Furthermore, among the states $\alpha \triangledown \beta \in \mathit{Inf}(\rho')$, the ones with the *longest tail* β (i.e., with maximal $|\beta|$), are exactly the ones where $\mathit{Inf}(\rho) = \widetilde{\beta}$. To make the latter statement somewhat more precise, define, for a given subset Q of the state space A', $\overline{Q} := \{\alpha \triangledown \beta \in Q \mid |\widetilde{\beta'}| \leq |\widetilde{\beta}|$ for all $\alpha' \triangledown \beta' \in Q\}$. Then one may show that

$$\overline{\mathit{Inf}(\rho')} = \{\alpha \triangledown \beta \in \mathit{Inf}(\rho') \mid \widetilde{\beta} = \mathit{Inf}(\rho)\}.$$

This shows how we can encode the success of runs of $\mathbb{A}$ in a parity condition for $\mathbb{A}'$. Putting

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

we ensure that for any word γ, we have the following equivalences:

$$\begin{array}{rcl} \mathbb{A} \text{ accepts } \gamma & \iff & \mathit{Inf}(\rho) \in \mathcal{M} \\ & \iff & \{\widetilde{\beta} \mid \alpha \triangledown \beta \in \overline{\mathit{Inf}(\rho')}\} \subseteq \mathcal{M} \\ & \iff & \max\{\Omega(\alpha \triangledown \beta) \mid \alpha \triangledown \beta \in \mathit{Inf}(\rho')\} \text{ is even} \\ & \iff & \mathbb{A}' \text{ accepts } \gamma. \end{array}$$

This suffices to prove the equivalence of $\mathbb{A}$ and $\mathbb{A}'$. QED

Example 4.18 With $\mathbb{A}_1$ the Muller automaton of Example 4.13, here are some examples of the transition function δ' of its parity equivalent $\mathbb{A}'$:

$$\begin{array}{rclcrcl} \delta'(a_b a_r a_g a_f a_0 \triangledown, b) & := & \triangledown a_r a_g a_f a_0 a_b & \quad & \delta'(\triangledown a_r a_g a_f a_0 a_b, b) & := & a_r a_g \triangledown a_0 a_b a_f \\ \delta'(a_b a_r a_g a_f a_0 \triangledown, r) & := & a_b a_r a_g a_f \triangledown a_0 & & \delta'(\triangledown a_r a_g a_f a_0 a_b, r) & := & \triangledown a_g a_f a_0 a_b a_r \\ \delta'(a_b a_r a_g a_f a_0 \triangledown, g) & := & a_b a_r a_g a_f \triangledown a_0 & & \delta'(\triangledown a_r a_g a_f a_0 a_b, g) & := & a_r \triangledown a_f a_0 a_b a_g \end{array}$$

Likewise, a few examples of the priority map:

$$\begin{array}{rcl} \Omega(a_b a_r a_g a_f \triangledown a_0) & := & 4 \\ \Omega(a_g a_f a_0 a_b \triangledown a_r) & := & 3 \\ \Omega(a_f a_r a_0 \triangledown a_b a_g) & := & 6 \\ \Omega(a_f a_0 \triangledown a_b a_r a_g) & := & 8 \end{array}$$

As the initial state of $\mathbb{A}'$, one could for instance take the sequence $a_r a_r a_g a_f a_0 \nabla$. $\lhd$

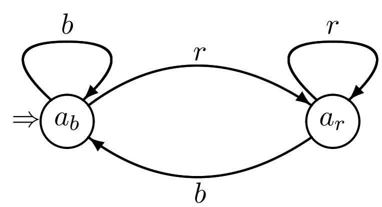
The following example shows that, in the case of deterministic stream automata, the recognizing power of Muller and parity automata is *strictly* stronger than that of Büchi automata.

Example 4.19 Consider the following language over the alphabet $C = \{b, r\}$:

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

That is, L consists of those C-streams that contain at most finitely many red items (that is, the symbol r occurs at most finitely often). We will give both a Muller and a parity automaton to recognize this language, and then show that there is no Büchi automaton for L.

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

For a parity automaton recognizing L, endow the diagram above with the priority map Ω_2 given by $\Omega_2(a_b) = 0$, $\Omega_2(a_r) = 1$. With this definition, there can only be one set of states of which the maximum priority is even, namely, the singleton $\{a_b\}$. Hence, this parity acceptance condition is the same as the Muller condition $\{\{a_b\}\}$.

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 $\alpha_1 = b^{n_0} rbbb\ldots$. 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, and so ρ_1 also passes through f_0 after n_0 steps. 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 $n_1 \in \omega$ such that ρ_1 passes through some state $f_1 \in F$ after $n_0 + n_1 + 1$ steps. Now consider the stream $\alpha_2 = b^{n_0} r b^{n_1} rbbb\ldots$, 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

$$\alpha_k = b^{n_0} r b^{n_1} \ldots r b^{n_k} rbbb\ldots \in L, \text{ for all } k. \tag{31}$$

Consider the stream

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

Containing infinitely many r's, α does not belong to L. Nevertheless, it follows from (31) that the run ρ of $\mathbb{A}$ on α passes through the states $f_0, f_1, \dots$ 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. $\lhd$

Remark 4.20 Since it is easy to see that the complement

$$\overline{L} = \{\alpha \in C^\omega \mid r \in \mathit{Inf}(\alpha)\}$$

of the language studied in Example 4.19 *is* recognized by a Büchi automaton, the example also shows that the class of Büchi recognizable stream languages is not closed under taking complementations. $\lhd$

4.3 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 4.21 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.* $\lhd$

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

Definition 4.22 Given a nondeterministic automaton $\mathbb{A} = \langle A, \Delta, Acc, a_I \rangle$, we define the relations $\mathtt{tr} \subseteq A \times C \times A$ and $\twoheadrightarrow \subseteq A \times C^* \times A$ in the obvious way: $a\mathtt{tr}ca'$ if $a' \in \Delta(a, c)$, $a \overset{\epsilon}{\twoheadrightarrow} a'$ if $a = a'$, and $a \overset{wc}{\twoheadrightarrow} a'$ if there is a a'' such that $a \overset{w}{\twoheadrightarrow} a''\mathtt{tr}ca'$. A *run* of a nondeterministic automaton $\mathbb{A} = \langle A, \Delta, Acc, a_I \rangle$ on an C-stream $\gamma = c_0c_1c_2\dots$ is an infinite A-sequence

$$\rho = a_0a_1a_2\dots$$

such that $a_0 = a_I$ and $a_i\mathtt{tr}c_ia_{i+1}$ for every $i \in \omega$. $\lhd$

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 make concerning the definition of the notion of acceptance.

Definition 4.23 A nondeterministic C-automaton $\mathbb{A} = \langle A, \Delta, Acc, a_I \rangle$ *accepts* a C-stream γ if there is a successful run of $\mathbb{A}$ on γ. $\lhd$

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 4.15, 4.16 and 4.17 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 4.24 *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 $M = \mathit{Inf}(\rho)$ 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.

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

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

We now turn to the *determinization* problem for stream automata. In the case of automata operating on finite words, it is not difficult to prove that nondeterminism does not really add recognizing power: any nondeterministic finite automaton $\mathbb{A}$ may be 'determinized', that is, transformed into an equivalent deterministic automaton $\mathbb{A}^d$.

Remark 4.25 Finite word automata (see Example 4.9) can be determinized by a fairly simple *subset construction.*

Let $\mathbb{A} = \langle A, \Delta, F, a_I \rangle$ be a nondeterministic finite word automaton. A run of $\mathbb{A}$ on a finite word $w = c_1 \cdots c_n$ is defined as a finite sequence $a_0 a_1 \cdots a_n$ such that $a_0 = a_I$ and $a_i \mathtt{trc}_i a_{i+1}$ for all $i < n$. $\mathbb{A}$ *accepts* a finite word w if there is a successful run, that is, a run $a_0 a_1 \cdots a_n$ ending in an accepting state a_n.

Given such a nondeterministic automaton, define a deterministic automaton $\mathbb{A}^+$ as follows. For the states of $\mathbb{A}^+$ we take the *macro-states* of $\mathbb{A}$, that is, the nonempty subsets of A. The deterministic transition function δ is given by

$$
\delta(P, c) := \bigcup_{a \in P} \Delta(a, c).
$$

In words, $\delta(P, c)$ consists of those states that can be reached from some state in P by making one a-step in $\mathbb{A}$. The accepting states of $\mathbb{A}^+$ are those macro-states that contain an accepting state from $\mathbb{A}$: $F^+ := \{P \in A^+ \mid P \cap F \neq \varnothing\}$, and its initial state is the singleton $\{a_I\}$.

In order to establish the equivalence of $\mathbb{A}$ and $\mathbb{A}^+$, we need to prove that for every word w, $\mathbb{A}$ has an accepting run on w iff the unique run of $\mathbb{A}^+$ on w is successful. The key claim in this proof is the following statement:

$$\widehat{\delta}(\{a_I\}, w) = \{a \in A \mid a_I \stackrel{w}{\twoheadrightarrow}_{\mathbb{A}} a\}. \tag{32}$$

stating that $\widehat{\delta}(\{a_I\}, w)$ consists of all the states that $\mathbb{A}$ can reach from a_I on input w. We leave the straightforward inductive proof of (32) as an exercise for the reader.

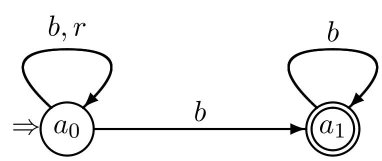
The equivalence of $\mathbb{A}$ and $\mathbb{A}^+$ then follows by the following chain of equivalences, for any finite word w: $\mathbb{A}^+$ accepts w iff $\widehat{\delta}(\{a_I\}, w) \in F^+$ iff $\widehat{\delta}(\{a_I\}, w) \cap F \neq \varnothing$ iff $a_I \stackrel{w}{\twoheadrightarrow}_{\mathbb{A}} a$ for some $a \in F$ iff $\mathbb{A}$ accepts w. $\lhd$

Unfortunately, the class of Büchi automata does not admit such a determinization procedure. As a consequence of Proposition 4.24 above, and witnessed by the Examples 4.19 and 4.26, the recognizing power of nondeterministic Büchi automata is strictly greater than that of their deterministic variants.

Example 4.26 For a nondeterministic Büchi automaton recognizing the language

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

of Example 4.19, 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, $F = \{a_1\}$. $\lhd$

There is positive news as well. A key result in automata theory states that when we turn to Muller and parity automata, nondeterminism does *not* increase recognizing power. This result follows from Proposition 4.24 and Theorem 4.27 below.

Theorem 4.27 *There is an effective procedure transforming a nondeterministic Büchi stream automaton into an equivalent deterministic Muller stream automaton.*

The *proof* of Theorem 4.27 will be given in the next section. As an important corollary we mention the following *Complementation Lemma.*

Proposition 4.28 *Let $\mathbb{A}$ be a nondeterministic Muller or parity automaton. Then there is an automaton $\overline{\mathbb{A}}$ of the same kind, such that $L_\omega(\overline{\mathbb{A}})$ is the complement of the language $L_\omega\mathbb{A}$.*

Leaving the proof of this proposition as an exercise for the reader, we finish this section with a summary of 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.

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

Having established these equivalences we naturally arrive at the following definition.

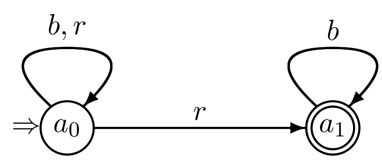
Definition 4.29 Let C be a finite set. A C-stream language $L \subseteq C^\omega$ is called *ω-regular* if there exists a C-stream automaton $\mathbb{A} = (A, \Delta, \Omega, a_I)$ such that $L = L_\omega(\mathbb{A})$, where $\mathbb{A}$ is either a (deterministic/nondeterministic) Muller or parity automaton, or a nondeterministic Büchi automaton. $\triangleleft$

4.4 Determinization of stream automata

This section is devoted to the proof of Theorem 4.27, which is based on a modification of the subset construction of Remark 4.25.

▶ `more information on determinization/simulation to be supplied`

Remark 4.30 This modification will have to be fairly substantial: As we will see now, Theorem 4.27 cannot be proved by a straightforward adaptation of the subset construction discussed in Remark 4.25. Consider the Büchi automaton $\mathbb{A}$ given by the following picture:



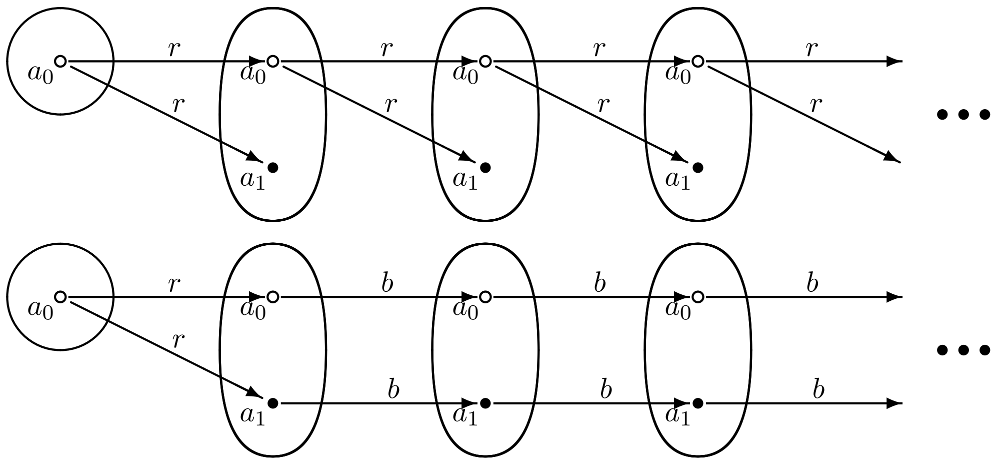
We leave it for the reader to verify that $L_\omega(\mathbb{A})$ consists of those streams of bs and rs that contain at least one and at most finitely many red items. In particular, the stream $r^\omega = rrrrr\ldots$ is rejected, while the stream $rb^\omega = rbbbb\ldots$ is accepted.

Now consider a deterministic automaton $\mathbb{A}^+$ of which the transition diagram is given by the subset construction. Then the run of the automaton $\mathbb{A}^+$ on r^ω is *identical* to its run on rb^ω:

$$a_0\{a_0, a_1\}\{a_0, a_1\}\{a_0, a_1\}\ldots$$

In other words, no matter which acceptance condition we give to $\mathbb{A}^+$, the automaton will accept either both r^ω and rb^ω, or neither. In either case $L_\omega(\mathbb{A}^+)$ will be different from $L_\omega(\mathbb{A})$.

As a matter of fact, it will be instructive to see in a bit more detail how the runs of $\mathbb{A}$ on r^ω and rb^ω, respectively, appear as 'traces' in the run of $\mathbb{A}^+$ on these two streams:



Clearly, where the second run contains one single trace that corresponds to a successful run of the automaton $\mathbb{A}$, in the first run, all traces that reach a successful state are aborted immediately. These two pictures clarify the subtle but crucial distinctions that get lost if we try to determinize via a straightforward subset construction. $\lhd$

In Safra's modification of the subset construction, the states of the deterministic automaton are *finite, structured collections of macro-states*; more specifically, if we order these macro-states by the inclusion relation we obtain a certain tree structure. The key idea underlying this modification is that at each step of the run, those elements of a macro-state that are accepting (i.e., members of the Büchi set of the original automaton), will be given some special treatment. Ultimately this enables one to single out the runs with a sequence of macro-states containing a good trace (that is, an infinite sequence of states constituting an accepting run of the nondeterministic automaton). Formally, we define these 'tree-ordered finite sets of macro-states' as Safra trees.

Definition 4.31 An ordered tree is a structure $\langle S, r, \lhd, <_H \rangle$ such that $\langle S, \lhd \rangle$ is a tree with root r; $\lhd$ is the 'child-of' relation, with $s \lhd t$ denoting that s is a child of t; and $<_H$ is a *sibling ordering relation*, that is, a strict partial order on S that totally orders the children of every node; if $s <_H t$ we may say that s is *older* than t. Given two nodes s and t, we say that s is *to the left of* t if there are nodes $s' <_H t'$ such that s and t are equal to or descend from s' and t', respectively.

A *Safra tree* over a set B is a pair (S, L) where S is a finite ordered tree, and $L : S \to \wp^+(B)$ is a labelling such that (i) for every node s, the set $\bigcup\{L(t) \mid t \lhd s\}$ is a *proper* subset of $L(s)$, and (ii) $L(s) \cap L(t) = \varnothing$ if s and t are siblings (i.e., distinct nodes with the same parent). $\lhd$

It is not hard to see that for any Safra tree (S, L) and for every state $b \in B$, b belongs to some label set of the tree iff it belongs to the label of the root. And, if b belongs to the label of the root, then there is a *unique* node $s \in S$, the so-called *lowest node of* b, such that $b \in L(s)$ but s has no child t with $b \in L(t)$. From these observations one easily derives that

$$|S| \leq |B|, \tag{33}$$

for every Safra tree over the set B.

We now turn to the details of the Safra construction.

Definition 4.32 Let $\mathbb{B}$ be a nondeterministic Büchi automaton $\mathbb{B} = \langle B, b_I, \Delta, F\rangle$. We will define a deterministic Muller automaton $\mathbb{B}^S = \langle B^S, a_I, \delta, \mathcal{M}\rangle$.

Assume that B has n states, and let $N := \{0, \ldots, 2n\}$; we will think of N as the set of *(potential) nodes* of a Safra tree. The carrier B^S will consist of the collection of all *colored Safra trees* over B, that is, all triples (S, L, θ) such that (S, L) is a Safra tree over B with $S \subseteq N$, and θ is a map coloring nodes of the tree either white or green. The initial state of $\mathbb{B}^S$ will be the Safra tree consisting of a single white node 0 labelled with the singleton $\{b_I\}$.

For the transition function on B^S, take an arbitrary colored Safra tree (S, L, θ). On input $c \in C$, the deterministic transition function δ on B^S transforms (S, L, θ) into a new colored, labelled Safra tree, by performing the sequence of actions below. (Note that at intermediate stages of this process, the structures may violate the conditions of Safra trees.)

1. *Separate accepting states* For each node $s \in S$ such that $L(s)$ contains accepting states, add a new[3] node $s' \in N \setminus S$ to S as the youngest child of s, and label s' with the set $L(s) \cap F$. (Such an s' can be canonically chosen as the smallest $n \in N$ such that $n \notin S$).

2. *Make macro-move* Apply the power set construction to the individual nodes: for each node s, replace its label $A \subseteq B$ with the set $\bigcup_{a \in A} \Delta(a, c)$.

3. *Merge traces* For each node s, remove those members from its label that already belong to the label of a state to the left of s (3a). After that remove all nodes with empty labels (3b).

4. *Mark successful nodes* For each (remaining) node s of which the label is *identical* to the union of the labels of its children, remove all proper descendants of s, and *mark* s by coloring it green. All other nodes are colored white.

For the Muller acceptance condition $\mathcal{M}$ of $\mathbb{B}^S$, put $M \in \mathcal{M}$ if there is some $s \in \{0, \ldots, 2n\}$ such that s is present as a node of every tree in M, and s is colored green in some tree in M. $\lhd$

Example 4.33 ▶ `Example to be supplied`

$\lhd$

▶ `discuss number of Safra trees`

It is obvious from the construction that $\mathbb{B}^S$ is a deterministic automaton, so what is left of the proof of Theorem 4.27 is to establish the equivalence of $\mathbb{B}$ and $\mathbb{B}^S$.

Proposition 4.34 *Let $\mathbb{B}$ be a nondeterministic Büchi automaton. Then*

$$L_\omega(\mathbb{B}) = L_\omega(\mathbb{B}^S).$$

[3]Observe that by (33) and the definition of N, there will always be sufficiently many nodes in N such that at least one element of N is left as a 'spare' node, possibly to be used at a later stage.

Proof.(Sketch) For the inclusion $\subseteq$, assume that there is a successful run $\rho = b_0 b_1 \ldots$ of $\mathbb{B}$ on some C-stream $\gamma = c_0 c_1 \ldots$. Consider the (unique) run $\sigma = (S_0, L_0, \theta_0)(S_1, L_1, \theta_1)\ldots$ of $\mathbb{B}^S$ on γ. Here each (S_i, L_i, θ_i) is a Safra tree with labeling L_i and coloring θ_i. We claim that there is an object s which after some initial phase belongs to each Safra tree of σ, and which is marked green infinitely often. The basic idea underlying the proof of this claim is to 'follow' the run ρ as a trace through the successive trees of σ.

First note that at every stage i, the state b_i of ρ belongs to the label $L_i(r_i)$ of the root r_i of the Safra tree S_i. It follows that the root always has a non-empty label, and hence it is never removed; thus we have $r_0 = r_1 = \ldots$, and so, with $r := r_0$, we have already found a node r such that r is present in every Safra tree in $\mathit{Inf}(\sigma)$. Now if r is colored green infinitely often, we are done.

So suppose that this is not the case. In other words, after a certain moment i, r will no longer be marked. Since $\rho = (b_i)_{i\in\omega}$ is by assumption a successful run of $\mathbb{B}$, it passes infinitely often through a successful state. Hence we may consider the first time $j > i$ for which b_j is an accepting state. But if $b_j \in F$, then in the next stage $j + 1$, first b_j is put in the label set of one of the children of r, and so after step 2 of that stage, the next state b_{j+1} of ρ belongs to the label set of one of the children, say, s_k, of the root. In step 3a, b_{j+1} may be removed from the label set of s_k, but only in case it was already present in the label set of an older sibling of s_k. It is not hard to see that in step 3b or 4, b_{j+1} will not be removed from the label set it belongs to.

We claim that in fact

$$\text{for all } k > j,\ b_k \in L_k(s_k),\ \text{for some child } s_k \text{ of } r. \tag{34}$$

The proof of this claim rests on the observation that b_k can only be removed from the set $\bigcup\{L_k(s) \mid s \lhd_k r\}$ in case r is a successful node in S_k, and we assumed that this was not the case. Now note that trace merges (as described in step 3a of the procedure) can cause states to be moved to the label set of a sibling, but only to an older one. Such a shift can thus only happen finitely often, so that after some stage j_1 there is a node s such that

$$\text{for all } k > j_1 : \quad s \in S_k, b_k \in L_k(s),\ \text{and } s \lhd_k r. \tag{35}$$

We can now repeat the argument with this s taking the role of r: either s itself is marked green infinitely often, or eventually, at some stage l, the ρ-state $b_l \in F$ will be placed at the next level, and remain there. Since the depth of the Safra trees involved is bounded, there must be some node s which after some initial phase belongs to each Safra tree in σ, and which is marked infinitely often.

For the opposite inclusion $\supseteq$, suppose that the (unique) run $\sigma = (S_0, L_0, \theta_0)(S_1, L_1, \theta_1)\ldots$ of $\mathbb{B}^S$ on γ is successful. Then by definition there is some node $s \in N = \{0, \ldots, 2n\}$ which after some initial phase will belong to each Safra tree in σ and which will subsequently be marked green infinitely often, say at the stages $k_1 < k_2 < \cdots$. For each $i > 0$, let A_i denote the macro-state of s at stage k_i, that is: $A_i := L_{k_i}(s)$.

Recall that γ is the infinite input stream $c_0 c_1 c_1 \cdots$. For natural numbers p and q, let $\gamma[p, q)$ denote the finite word $c_p \cdots c_{q-1}$, so that γ is equal to the infinite concatenation

$$\gamma = \gamma[0, k_1) \cdot \gamma[k_1, k_2) \cdot \gamma[k_2, k_3) \cdots$$

Since our construction is a refinement of the standard subset construction of Remark 4.25, by (32) it easily follows from the definitions of δ that for every state $a \in A_1$ there is a $\gamma[0,k_1)$-labeled path from b_I to a, or briefly:

$$\text{for all } a \in A_1 \text{ we have } b_I \stackrel{\gamma[0,k_1)}{\twoheadrightarrow} a. \tag{36}$$

With a little more effort, crucially involving the conditions for marking nodes, and the rules governing the creation and maintenance of nodes, one may prove that

$$\text{for all } i > 0 \text{ and for all } a \in A_{i+1} \text{ there is an } a' \in A_i \text{ such that } a' \stackrel{\gamma[k_i,k_{i+1})}{\twoheadrightarrow_F} a. \tag{37}$$

Here $a' \stackrel{\gamma[k_i,k_{i+1})}{\twoheadrightarrow_F} a$ means that there is a $\gamma[k_i,k_{i+1})$-labelled path from a' to a which passes through some state in F. Details of this proof are left as an exercise to the reader.

The remainder of the proof consists of finding a successful run of $\mathbb{B}$ on γ as the concatenation of a run segment given by (36) and infinitely many run segments given by (37). For this we use König's Lemma.

Defining $A_0 := \{b_I\}$, we will construct a tree, all of whose nodes are pairs of the form (a,i) with $a \in A_i$. As the (unique) parent of a node $(a, i+1)$ we pick one of the pairs (a', i) given by (36) and (37), respectively. Obviously this is a well-formed, infinite, finitely branching tree. So by König's Lemma, there is an infinite branch $(a_0, 0)(a_1, 1)\cdots$. By construction, we have $a_0 = b_I$, while for each $i \geq 0$ there is a $\gamma[k_i,k_{i+1})$-labelled path in $\mathbb{B}$ from a_i to a_{i+1} which passes through some accepting state of $\mathbb{B}$. The infinite concatenation of these paths gives a run of $\mathbb{B}$ on γ, which visits infinitely often an accepting state of $\mathbb{B}$, and hence by finiteness of B, it visits some state of $\mathbb{B}$ infinitely often. Clearly then this run is accepting. QED

4.5 Logic and automata

- ▶ `discuss the relation between stream automata, the linear` μ`-calculus, and monadic second-order logic;`
- ▶ `discuss linear time logic`

4.6 A coalgebraic perspective

In this section we introduce a coalgebraic perspective on stream automata. We have two reasons for doing so. First, we hope that this coalgebraic presentation will facilitate the introduction, further on, of automata operating on different kinds of structures. 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.

In this context, it makes sense to consider a slightly wider class than streams only.

Definition 4.35 A *C-flow* is a pair $\mathbb{S} = \langle S, \sigma\rangle$ with $\sigma : S \to C \times S$. Often we will write $\sigma(s) = (\sigma_C(s), \sigma_0(s))$. If we single out an (initial) state $s_0 \in S$ in such a structure, we obtain a *pointed C-flow* $(\mathbb{S}, s_0)$. ◁

Example 4.36 Streams over an alphabet C can be seen as pointed C-flows: simply identify the word $\gamma = c_0c_1c_2\ldots$ with the pair $(\langle\omega, \lambda n.(c_n, n+1)\rangle, 0)$. Conversely, with any pointed flow $\langle\mathbb{S}, s\rangle$ we may associate a unique stream $\gamma_{\mathbb{S},s}$ by inductively defining $s_0 := s$, $s_{i+1} := \sigma_0(s_i)$, and putting $\gamma_{\mathbb{S}}(n) := \sigma_C(s_n)$. $\lhd$

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

Definition 4.37 Let $\mathbb{S}$ and $\mathbb{S}'$ be two C-flows. Then a nonempty relation $Z \subseteq S \times S'$ is a *bisimulation* if the following holds, for every $(s, s') \in Z$:

(color) $\sigma_C(s) = \sigma'_C(s')$;

(successor) $(\sigma_0(s), \sigma'_0(s')) \in Z$.

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

As an example, it is not hard to see that any pointed flow $(\mathbb{S}, s)$ is bisimilar to the stream $\gamma_{\mathbb{S},s}$ that we may associate with it (see Example 4.36). Restricted to the class of streams, bisimilarity means *identity*.

Definition 4.38 A stream is called *regular* if it is bisimilar to a finite pointed flow. $\lhd$

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

Now on the one hand make a restriction in the sense that the classifying flow 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

$$\Delta : A \to \wp(C \times A). \tag{38}$$

Remark 4.39 This presentation (38) of nondeterminism 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 4.21 as the transition structure of nondeterministic automata:

$$A \to \wp(C \times A) \;\cong\; (A \times C) \to \wp(A). \tag{39}$$

To see why this is so, an easy proof suffices. Using the principle of currying 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 (39) using the bijection between subsets and their characteristic functions.

Concretely, we may identify a map $\Delta : (A \times C) \to \wp(A)$ with the map $\Delta' : A \to \wp(C \times A)$ given by

$$\Delta'(a) := \{(c, a') \mid a' \in \Delta(a, c)\}. \tag{40}$$

$\lhd$

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 flow. Apart from this, an obvious difference with flows is that stream automata also have an acceptance condition.

Definition 4.40 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 flow 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 flows can itself be captured game-theoretically, using a trivialized version of the bisimilarity game for Kripke models of Definition 1.25. Consider two flows $\mathbb{A}$ 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, $\exists$ loses immediately; if on the other hand $\alpha_C(a) = \sigma_C(s)$, 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
$(a, s) \in A \times S$	$\exists$	$\{(\alpha_0(a), \sigma_0(s)) \mid \alpha_C(a) = \sigma_C(s)\}$

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
$(a, s) \in A \times S$	-	$\{(\alpha(a), s)\}$
$(\alpha, s) \in (C \times A) \times S$	$\exists$	$\{(\alpha_0, \sigma_0(s)) \mid \alpha_C = \sigma_C(s)\}$

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

Definition 4.41 Given a nondeterministic C-stream automaton $\mathbb{A} = \langle A, a_I, \Delta, Acc\rangle$ and a pointed flow $\mathbb{S} = \langle S, s_0, \sigma\rangle$, we now define the *acceptance game* $\mathcal{A}(\mathbb{A}, \mathbb{S})$ as the following board game.

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

Table 7: Acceptance game for nondeterministic stream automata

Its positions and rules are given in Table 7, 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' $a_0 a_1 a_2 \dots$ of this sequence belongs to Acc. $\lhd$

Definition 4.42 A nondeterministic C-stream automaton $\mathbb{A} = \langle A, a_I, \Delta, Acc\rangle$ *accepts* a pointed flow $\mathbb{S} = \langle S, s_0, \sigma\rangle$ if the pair (a_I, s_0) is a winning position for $\exists$ in the game $\mathcal{A}(\mathbb{A}, \mathbb{S})$. $\lhd$

The following proposition states that the two ways of looking at nondeterministic automata are equivalent.

Proposition 4.43 *Let $\mathbb{A} = \langle A, a_I, \Delta, Acc\rangle$, with $\Delta : (A \times C) \to \wp(A)$ be a nondeterministic C-automaton, and let $\mathbb{A}'$ be the nondeterministic C-stream automaton $\langle A, a_I, \Delta', Acc\rangle$, where $\Delta' : A \to \wp(C \times A)$ is given by (40). Then $\mathbb{A}$ and $\mathbb{A}'$ are equivalent.*

In the sequel we will *identify* the two kinds of nondeterministic automata, speaking of the *coalgebraic presentation* $\langle A, a_I, \Delta' : A \to \wp(C \times A), Acc\rangle$ of an automaton $\langle A, a_I, \Delta : (A \times C) \to \wp(A), Acc\rangle$.

Notes

The idea to use finite automata for the classification of infinite words originates with Büchi. In [6] he used stream automata with (what we now call) a Büchi acceptance condition to prove the decidability of the second-order theory of the natural numbers (with the successor relation). In the subsequent development of the theory of stream automata, other acceptance conditions were introduced. The Muller condition is named after the author of [21]. The invention of the parity condition, which can be seen as a refinement of the Rabin condition, is usually attributed to Emerson & Jutla [11], Mostowski [20], and/or Wagner.

The first construction of a deterministic equivalent to a nondeterministic Muller automaton was given by McNaughton [18]. The construction we presented in section 4.4 is due to Safra [26]. Finally, the coalgebraic perspective on stream automata presented in the final section of this chapter is the author's.

Exercises

Exercise 4.1 Provide Büchi automata recognizing exactly the following stream languages:

(a) $L_a = \{\alpha \in \{a, b, c\}^\omega \mid a \text{ and } b \text{ occur infinitely often in } \alpha\}$

(b) $L_b = \{\alpha \in \{a, b, c\}^\omega \mid \text{any } a \text{ in } \alpha \text{ is eventually followed by a } b\}$

(c) $L_c = \{\alpha \in \{a, b\}^\omega \mid \text{between any two } a\text{'s is an even number of } b\text{'s}\}$

(d) $L_d = \{\alpha \in \{a, b, c\}^\omega \mid ab \text{ and } cc \text{ occur infinitely often in } \alpha\}$

Exercise 4.2 Let C be a finite set. Show that the class of ω-regular languages over C is closed under the Boolean operations, i.e., show that

(a) If $L \subseteq C^\omega$ is ω-regular then its complement $\overline{L} := \{\gamma \in C^\omega \mid \gamma \notin L\}$ is ω-regular.

(b) If L_1 and L_2 are ω-regular C-stream languages, then $L_1 \cup L_2$ is ω-regular.

(c) If L_1 and L_2 are ω-regular C-stream languages, then $L_1 \cap L_2$ is ω-regular.

Exercise 4.3 Observe that Büchi automata can also be seen as finite automata operating on *finite* words (see Example 4.9.

(a) Show the following, for any deterministic Büchi automaton $\mathbb{A}$:

$$L_\omega(\mathbb{A}) = \{\alpha \in \Sigma^\omega \mid \text{infinitely many prefixes of } \alpha \text{ belong to } L(\mathbb{A})\}.$$

(b) Does this hold for nondeterministic Büchi automata as well?

Exercise 4.4 Let C and D be finite sets and let $f : C \to D$ be a function. The function f can be extended to a function $\overline{f} : C^\omega \to D^\omega$ in the obvious way by putting $\overline{f}(\gamma) := f(c_0)f(c_1)f(c_2)\ldots \in D^\omega$ for any C-stream $\gamma \in C^\omega$. For a given C-stream language $L \subseteq C^\omega$ we define

$$\overline{f}(L) := \{\overline{f}(\gamma) \mid \gamma \in L\} \subseteq D^\omega.$$

(a) Show that $L \subseteq C^\omega$ is ω-regular implies $f(L) \subseteq D^\omega$ is ω-regular.

(b) Show that there is a C-stream language $L \subseteq C^\omega$ such that $L = L_\omega(\mathbb{A})$ for some *deterministic* Büchi automaton $\mathbb{A}$ and such that $f(L) \subseteq D^\omega$ is not recognizable by any deterministic Büchi automaton.

Exercise 4.5 Prove that nondeterministic Büchi automata have the same recognizing power as their Muller variants by showing that the automata $\mathbb{A}'$ and $\mathbb{A}$ in the proof of Proposition 4.24 are indeed equivalent.

Exercise 4.6 Consider the language L_d of exercise 4.1.

(a) Give a clear description of the complement $\overline{L_d}$ of L_d.

(b) Give a nondeterministic Büchi automaton recognizing exactly the language $\overline{L_d}$.

(c) Prove that there is no deterministic Büchi automaton recognizing the language $\overline{L_d}$. (Hint: use the theorem from Exercise 4.3.)

Exercise 4.7 Provide deterministic Muller automata recognizing the following languages:

(a) L_d of exercise 4.1.

(b) $L_a = \{\alpha \in \{a, b, c\}^\omega \mid$ between every pair of a's is an occurrence of bb or cc $\}$.

Exercise 4.8 (regularity) Let C be a finite set, and let $L \subseteq C^\omega$ be a stream language over C. Prove that if L is ω-regular, then it contains a stream of the form uv^ω where $u \in C^*$ and $v \in C^+$.

Exercise 4.9 Describe the languages that are recognized by the following Muller automata (presented in tabular form, with $\Rightarrow$ indicating the initial state):

(a)

$\mathbb{A}$	a	b
$\Rightarrow\ q_0$	q_1	q_2
q_1	q_0	q_2
q_2	q_1	q_0

with $\mathcal{F} := \{\{q_0, q_1\}, \{q_0, q_2\}\}$.

(b) The same automaton as in (a) but with $\mathcal{F} := \{\{q_1, q_2\}, \{q_0, q_1, q_2\}\}$.

(c)

$\mathbb{A}$	a	b	c
$\Rightarrow\ q_0$	q_1	q_0	q_0
q_1	q_0	q_2	q_0
q_2	q_0	q_0	q_3
q_3	q_0	q_0	q_0

with $\mathcal{F} := \{\{q_0\}, \{q_0, q_1\}, \{q_0, q_1, q_2\}\}$.

Exercise 4.10 Prove (37) in the proof of Proposition 4.34. That is, show that

$$\text{for all } i > 0 \text{ and for all } a \in A_{i+1} \text{ there is an } a' \in A_i \text{ such that } a' \overset{\gamma[k_i,k_{i+1})}{\twoheadrightarrow_F} a.$$

Can you also prove that, conversely,

$$\text{for all } i > 0 \text{ and for all } a \in A_i \text{ there is an } a' \in A_{i+1} \text{ such that } a' \overset{\gamma[k_i,k_{i+1})}{\twoheadrightarrow_F} a?$$

5 Parity 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.

5.1 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 5.1 If $\Pi \in \{0,1\}$ is a player, then $\overline{\Pi}$ denotes the *opponent* $1 - \Pi$ of Π. $\lhd$

A board game is played on a *board* or *arena*, which is nothing but a directed graph in which each node is marked with either 0 or 1. A *match* or *play* of the game consists of the two players moving a pebble or 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 5.2 A *board* or *arena* is a structure $\mathbb{B} = \langle B_0, B_1, E\rangle$, such that B_0 and B_1 are disjoint sets, 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* or *legitimate 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 Π_p denote the (unique) player such that $p \in B_{\Pi_p}$, and say that *p belongs to* Π_p, or that it is Π_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. We refer to the Appendix A for some notation concerning paths.

Definition 5.3 A *path* through a board $\mathbb{B} = \langle B_0, B_1, E\rangle$ is a (finite or infinite) sequence $\Sigma \in B^\infty$ such that $E\Sigma_i\Sigma_{i+1}$ whenever applicable. A *full* or *complete match* or *play* through $\mathbb{B}$ is either an infinite $\mathbb{B}$-path, or a finite $\mathbb{B}$-path Σ ending with a dead end (i.e. $E[last(\Sigma)] = \varnothing$).

A *partial match* is a finite path through $\mathbb{B}$ that is not a full match; in other words, the last position of a partial match is not a dead end. We let PM_Π denote the set of partial matches such that Π 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 Π_Σ; that is, $\Pi_\Sigma := \Pi_{last(\Sigma)}$. $\lhd$

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.

The importance of this explains the definition of the notion of a *subboard*. Note that any set of positions on a board naturally induces a board of its own, based on the restricted edge relation. We will only call this structure a subboard, however, if there is no disagreement between the two boards when it comes to players being stuck or not.

Definition 5.4 Given a board $\mathbb{B} = \langle B_0, B_1, E\rangle$, a subset $A \subseteq B$ determines the following board $\mathbb{B}_A := \langle A \cap B_0, A \cap B_1, E_{\restriction A}\rangle$, where $E_{\restriction A} := E \cap (A \times A)$ is the *restriction* of E to A. This structure is called a *subboard* of $\mathbb{B}$ if for all $p \in A$ it holds that $E[p] = \varnothing$ iff $E_{\restriction A}[p] = \varnothing$. $\lhd$

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 5.5 Given a board $\mathbb{B}$, a *winning condition* is a map $W : B^\omega \to \{0, 1\}$. An infinite match Σ is *won* by $W(\Sigma)$. 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 5.6 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 Σ, its first element *first*(Σ) is called the *starting position* of the match. We let $\mathrm{PM}_\Pi(q)$ denote the set of partial matches for Π 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 method that the player uses to decide how to continue partial matches when it is their turn to move. More precisely, a strategy is a function mapping partial plays for the player to new positions. It is a matter of definition whether one requires a strategy to always assign moves that are legitimate, or not; here we will not make this requirement.

Definition 5.7 Given a board game $\mathcal{G} = \langle B_0, B_1, E, W\rangle$ and a player Π, a Π-*strategy*, or a *strategy for* Π, is a map $f : \mathrm{PM}_\Pi \to B$. In case we are dealing with an initialized game $\mathcal{G}@q$, then we may take a strategy to be a map $f : \mathrm{PM}_\Pi(q) \to B$. A match Σ is *consistent with* or *guided by* a Π-strategy f if for any $\Sigma' \sqsubset \Sigma$ with *last*$(\Sigma') \in B_\Pi$, the next position on Σ (after Σ') is indeed the element $f(\Sigma')$.

A Π-strategy f is *surviving* in $\mathcal{G}@q$ if the moves that it prescribes to f-guided partial matches in $\mathrm{PM}_\Pi@p$ are always admissible to Π, and *winning for* Π in $\mathcal{G}@p$ if in addition all f-guided full matches starting at p are won by Π. A position $q \in B$ is *winning for* Π if Π has a winning strategy for the game $\mathcal{G}@q$; the collection of all winning positions for Π in $\mathcal{G}$ is called the *winning region for* Π *in* $\mathcal{G}$, and denoted as $\mathrm{Win}_\Pi(\mathcal{G})$. $\lhd$

Intuitively, f being a surviving strategy in $\mathcal{G}@q$ means that Π never gets stuck in an f-guided match of $\mathcal{G}@q$, and so guarantees that Π can stay in the game forever.

Convention 5.8 Observe that we allow strategies that prescribe illegitimate moves. In practice, it will often be convenient to extend the definition of a strategy even further to include maps f that are *partial* in the sense that they are only defined on a proper subset of PM_{Π}. We will only permit ourselves such a sloppiness if we can guarantee that $f(\Sigma)$ is defined for every $\Sigma \in \mathrm{PM}_{\Pi}$ that is consistent with the partial Π-strategy f, so that the situation where the partial strategy actually would fail to suggest a move, will never occur.

It is easy to see that a position in a game $\mathcal{G}$ cannot be winning for *both* players. On the other hand, the question whether a position p is always a winning position for *one* of the players, is a rather subtle one. Observe that in such games the two winning regions *partition* the game board.

Definition 5.9 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. $\lhd$

It turns out that the axiom of choice implies the existence of infinite games that admit positions from which neither player has a winning strategy.

▶ `Add some more detail, including a remark on the axiom of determinacy in set theory.`

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. Particularly nice are so-called *positional* strategies, which only depend on the current position (i.e., the final position of the partial play). Although their importance is sometimes overrated, positional strategies are convenient to work with, and they will be critically needed in the proofs of some of the most fundamental results in the automata-theoretic approach to fixpoint logic.

Definition 5.10 A strategy f is *positional* or *history-free* if $f(\Sigma) = f(\Sigma')$ for any Σ, Σ' with $\mathit{last}(\Sigma) = \mathit{last}(\Sigma')$. $\lhd$

Convention 5.11 A positional Π-strategy may be represented as a map $f : B_{\Pi} \to B$.

As a slight generalisation of positional strategies, *finite-memory strategies* can be computed using only a finite amount of information about the history of the match. More details can be found in Exercise 5.2.

5.2 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 5.12 A *coloring* of B is a function $\Gamma : B \to C$ assigning to each position $p \in B$ a *color* $\Gamma(p)$ taken from some finite set C of colors. By putting $\Gamma(p_0 p_1 \cdots) := \Gamma(p_0)\Gamma(p_1)\cdots$ we can naturally extend such a coloring $\Gamma : B \to C$ to a map $\Gamma : B^\omega \to C^\omega$. $\lhd$

Now if $\Gamma : B \to C$ is a coloring, for any infinite sequence $\Sigma \in B^\omega$, the map $\Gamma \circ \Sigma \in C^\omega$ 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 5.13 Let $\mathbb{B}$ be a board and $\Gamma : B \to C$ a coloring of B. Given an infinite sequence $\Sigma \in B^\omega$, we let $\mathit{Inf}_\Gamma(\Sigma)$ denote the set of colors that occur infinitely often in the sequence $\Gamma \circ \Sigma$.

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 $W_\mathcal{M} : B^\omega \to \{0, 1\}$:

$$W_\mathcal{M}(\Sigma) := \begin{cases} 0 & \text{if } \mathit{Inf}_\Gamma(\Sigma) \in \mathcal{M} \\ 1 & \text{otherwise.} \end{cases}$$

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

In words, player 0 wins an infinite match $\Sigma = p_0 p_1 \cdots$ 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. These results can in fact be generalised to arbitrary *regular games*, that is, board games where the winning condition is given as an ω-regular language over some colouring of the board. We refer to Exercise 5.2) for more details.

These results becomes even nicer if the Muller condition allows a formulation in terms of a *priority 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 5.14 Let B be some set; a *priority 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

$$W_\Omega(\Sigma) \ := \ \max(\mathit{Inf}_\Omega(\Sigma)) \mod 2.$$

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

The key property that makes parity games so interesting is that they enjoy positional determinacy. We will prove this in section 5.4. First we turn to a special case, viz., the reachability games.

5.3 Reachability Games

Reachability games are a special kind of board games. They are played on a board such as described in section 5.1, but now we also choose a subset $A \subseteq B$. The aim of the game is for the one player to move the pebble into A and for the other to avoid this to happen.

Definition 5.15 Fix a board $\mathbb{B}$ and a subset $A \subseteq B$. The reachability game $\mathcal{R}_\Pi(\mathbb{B}, A)$ is then defined as the game over $\mathbb{B}$ in which Π wins as soon as a position in A is reached or if $\overline{\Pi}$ gets stuck. On the other hand, $\overline{\Pi}$ wins if he can manage to keep the token outside of A infinitely long, or if Π gets stuck. $\lhd$

As an example, if $A = \varnothing$, in order to win the game $\mathcal{R}_\Pi(\mathbb{B}, A)$ for player $\overline{\Pi}$ it simply suffices to stay alive forever, while Π can only win by forcing $\overline{\Pi}$ to get stuck.

Remark 5.16 If we want reachability games to fit the format of a board game exactly, we have to modify the board, as follows. Given a reachability game $\mathcal{R}_\Pi(\mathbb{B}, A)$, define the board $\mathbb{B}' := \langle B'_0, B'_1, E' \rangle$ by putting:

$$
\begin{array}{lcl}
B'_\Pi & := & B_\Pi \setminus A \\
B'_{\overline{\Pi}} & := & B_{\overline{\Pi}} \cup A \\
E' & := & \{(p, q) \in E \mid p \notin A\}.
\end{array}
$$

In other words, $\mathbb{B}'$ is like $\mathbb{B}$ except that player $\overline{\Pi}$ gets stuck in a position belonging to A. Furthermore, the winning conditions of such a game are very simple: simply define $W : B^\omega \to \{0, 1\}$ as the constant function mapping all infinite matches to $\overline{\Pi}$. This can easily be formulated as a parity condition. $\lhd$

Since reachability games can thus be formulated as very simple parity games, the following theorem, stating that reachability games enjoy positional determinacy, can be seen as a warming up exercise for the general case. We leave the proof of this result as an exercise for the reader.

Theorem 5.17 (Positional determinacy of reachability games) *Let $\mathcal{R}$ be a reachability game. Then there are positional strategies f_0 and f_1 for 0 and 1, respectively, such that for every position q there is a player Π such that f_Π is a winning strategy for Π in $\mathcal{R}@q$.*

Definition 5.18 The winning region for Π in $\mathcal{R}_\Pi(\mathbb{B}, A)$ is called the *attractor set* of Π for A in $\mathbb{B}$, notation: $Attr^{\mathbb{B}}_\Pi(A)$. In the sequel we will fix a positional winning strategy for Π in $\mathcal{R}_\Pi(\mathbb{B}, A)$ and denote it as $attr^{\mathbb{B}}_\Pi(A)$. $\lhd$

Note that Π-attractor sets always contain all points from which Π can make sure that $\overline{\Pi}$ gets stuck. Furthermore, it is easy to see that in $attr_\Pi(A)$-guided matches the pebble never leaves $Attr_\Pi(A)$ (at least if the match starts inside $Attr_\Pi(A)$!).

Proposition 5.19 *$Attr_\Pi$ is a closure operation on $\mathcal{P}(B)$, i.e.*

1. *$A \subseteq A'$ implies $Attr_\Pi(A) \subseteq Attr_\Pi(A')$,*
2. *$A \subseteq Attr_\Pi(A)$,*
3. *$Attr_\Pi(Attr_\Pi(A)) = Attr_\Pi(A)$.*

A kind of counterpart to attractor sets are *traps*. In words, a set A is a Π-trap if Π can't get the pebble out of A, while her opponent has the power to keep it inside A.

Definition 5.20 Given a board $\mathbb{B}$, we call a subset $A \subseteq B$ a Π-*trap* if $E[b] \subseteq A$ for all $b \in A \cap B_\Pi$, while $E[b] \cap A \neq \varnothing$ for all $b \in A \cap B_{\overline{\Pi}}$. $\triangleleft$

Note that a Π-trap does not contain $\overline{\Pi}$-endpoints and that $\overline{\Pi}$ will therefore never get stuck in a Π-trap. We conclude this section with a useful proposition.

Proposition 5.21 *Let $\mathbb{B}$ be a board and $A \subseteq B$ an arbitrary subset of B. Then the following assertions hold.*

1. *If A is a Π-trap then A is a subboard of B.*
2. *The union $\bigcup\{A_i \mid i \in I\}$ of an arbitrary collection of Π-traps is again a Π-trap.*
3. *If A is a Π-trap then so is $Attr_{\overline{\Pi}}(A)$.*
4. *The complement of $Attr_\Pi(A)$ is a Π-trap.*
5. *If A is a Π-trap in $\mathbb{B}$ then any $C \subseteq A$ is a Π-trap in $\mathbb{B}$ iff C is a Π-trap in $\mathbb{B}_A$.*

Proof. All statements are easily verified and thus the proof is left to the reader. QED

5.4 Positional Determinacy of Parity Games

Theorem 5.22 (Positional Determinacy of Parity Games) *For any parity game $\mathcal{G}$ there are positional strategies f_0 and f_1 for 0 and 1, respectively, such that for every position q there is a player Π such that f_Π is a winning strategy for Π in $\mathcal{G}@q$.*

We start with the definition of players' paradises. In words, a subset $A \subseteq B$ is a Π-paradise if Π has a positional strategy f which guarantees her both that she wins the game, and that the token stays in A.

Definition 5.23 Given a parity game $\mathcal{G}(\mathbb{B}, \Omega)$, we call a $\overline{\Pi}$-trap A a Π-*paradise* if there exists a positional winning strategy $f : A \cap B_\Pi \to A$. $\triangleleft$

The following proposition establishes some basic facts about paradises.

Proposition 5.24 *Let $\mathcal{G}(\mathbb{B}, \Omega)$ be a parity game. Then the following assertions hold:*

1. *The union* $\bigcup\{P_i \mid i \in I\}$ *of an arbitrary set of* Π*-paradises is again a* Π*-paradise.*

2. *There exists a largest* Π*-paradise.*

3. *If* P *is a* Π*-paradise then so is* $Attr_{\Pi}(P)$*.*

Proof. The main point of the proof of part (1) is that we somehow have to uniformly choose a strategy on the intersection of paradises, such that we will end up following the strategy of only one paradise. For this purpose, we assume that we have a well-ordering on the index set I (i.e., for the general case we assume the Axiom of Choice).

For the details, assume that $\{P_i \mid i \in I\}$ is a family of paradises, and let f_i be the positional winning strategy for P_i. Note that $P := \bigcup\{P_i \mid i \in I\}$ is a trap for $\overline{\Pi}$ by Proposition 5.21. Assume that $<$ is a well-ordering of I, so that for each $q \in P$ there is a *minimal* index $\min(q)$ such that $q \in P_{\min(q)}$. Define a positional strategy on P by putting

$$f(q) := f_{\min(q)}(q).$$

This strategy ensures at all times that the pebble either stays in the current paradise, or else it moves to a paradise of lower index, and so, any match where Π plays according to f will proceed through a sequence of Π-paradises of decreasing index. Because of the well-ordering, this decreasing sequence of paradises cannot be strictly decreasing, and thus we know that after finitely many steps the pebble will remain in the paradise where it is, say, P_j. From that moment on, the match is continued as an f_j-guided match inside P_j, and since f_j is by assumption a winning strategy when played inside P_j, this match is won by Π.

Part (2) of the proposition should now be obvious: clearly the union of all Π-paradises is the greatest Π-paradise.

In order to prove part (3) we need to show that there exists a winning strategy for Π. The principal idea is to first move to P by $attr_{\Pi}(P)$ and once there to follow the winning strategy in P. Let f' be the winning strategy for P, we then define the following strategy f on $Attr_{\Pi}(P)$ by

$$f(p) := \begin{cases} f'(p) & \text{if } p \in P \\ attr_{\Pi}(P)(p) & \text{otherwise.} \end{cases}$$

A match consistent with this strategy will stay in $Attr_{\Pi}(P)$ because it is a $\overline{\Pi}$-trap and $f(p) \in Attr_{\Pi}(P)$ for all $p \in Attr_{\Pi}(P)$. It is winning because if ever the match arrives at a point $p \in P$ then play continues as if the match were completely in P; and since f' was supposed to be a winning strategy for Π this play is won by Π. However if we start outside P we will at first follow the strategy $attr_{\Pi}(P)$ which will ensure that Π either wins or that the pebble ends up in P, in which case Π will also win. QED

Now we are ready to prove the main assertion from which Theorem 5.22 immediately follows.

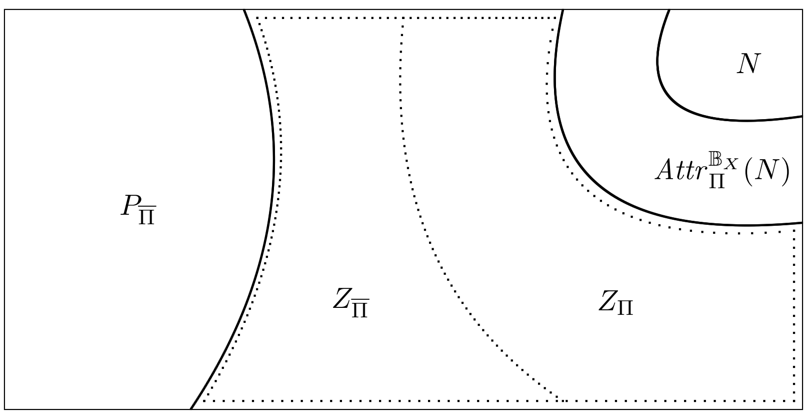
Proposition 5.25 *The board of a parity game* $\mathcal{G}(\mathbb{B}, \Omega)$ *can be partitioned into a 0-paradise and a 1-paradise.*

Proof. We will prove this proposition by induction on n, the maximal parity in the game (i.e. $n = \max(\Omega[B])$). If $n = 0$ we are dealing with a reachability game (namely $\mathcal{R}_1(\mathbb{B}, \varnothing)$), and from the results in section 5.3 we may derive that $Attr_1(\varnothing)$ is a 1-paradise and its complement is a 0-paradise. So the proposition holds in case $n = 0$.

Therefore in the remainder we can assume that $n \geq 1$. Let $\Pi := n \mod 2$, that is, Π wins an infinite play Σ if $\max(Inf(\Sigma)) = \max(\Omega[B]) = n$. Let $P_{\overline{\Pi}}$ be the maximal $\overline{\Pi}$-paradise, with associated positional strategy f. It now suffices to show that $X := B \setminus P_{\overline{\Pi}}$ is a Π-paradise.

First we shall show that X is a $\overline{\Pi}$-trap. By proposition 5.24(3) it follows that $Attr_{\overline{\Pi}}(P_{\overline{\Pi}})$ is itself also a $\overline{\Pi}$-paradise. By maximality of $P_{\overline{\Pi}}$ and the fact that $Attr_{\overline{\Pi}}$ is a closure operation, it follows that $P_{\overline{\Pi}} = Attr_{\overline{\Pi}}(P_{\overline{\Pi}})$. Thus by Proposition 5.21(4) we see that X, being the complement of a $\overline{\Pi}$-attractor set is a $\overline{\Pi}$-trap.

Consider $\mathcal{G}_X$, the subgame of $\mathcal{G}$ restricted to X. Note that by proposition 5.21(1), X is a subboard of $\mathbb{B}$, so the name 'subgame' is justified. Define $N := \{b \in X \mid \Omega(b) = n\}$ to be the set of all points in X with priority n and let $Z := X \setminus Attr_{\Pi}^{\mathbb{B}_X}(N)$. Since Z is the complement of a Π-attractor set in $\mathbb{B}_X$ it is a Π-trap in $\mathbb{B}_X$ and hence a Π-trap of $\mathbb{B}$.



By the induction hypothesis we can split the subgame $\mathcal{G}_Z$ into a 0-paradise Z_0 and a 1-paradise Z_1, see the picture. The winning strategies in these paradises we call f_0 and f_1 respectively. (All notions are with regard to the game $\mathcal{G}_Z$.) We want to show that $Z_{\overline{\Pi}} = \varnothing$, so that $Z = Z_{\Pi}$.

To this aim, we claim that $P_{\overline{\Pi}} \cup Z_{\overline{\Pi}}$ is a $\overline{\Pi}$-paradise in $\mathcal{G}$, and in order to prove this, we consider the following strategy g of $\overline{\Pi}$:

$$g(b) := \begin{cases} f(b) & \text{if } b \in P_{\overline{\Pi}} \\ f_{\overline{\Pi}}(b) & \text{if } b \in Z_{\overline{\Pi}}. \end{cases}$$

It is left as an exercise for the reader to show that this is indeed a positional winning strategy for $\overline{\Pi}$ in $\mathcal{G}$, and in addition it keeps the pebble inside $P_{\overline{\Pi}} \cup Z_{\overline{\Pi}}$. By the definition of $P_{\overline{\Pi}}$ as the maximal $\overline{\Pi}$-paradise, we see that $P_{\overline{\Pi}} = P_{\overline{\Pi}} \cup Z_{\overline{\Pi}}$ and since $P_{\overline{\Pi}}$ and $Z_{\overline{\Pi}}$ are disjoint we conclude that $Z_{\overline{\Pi}}$ is empty indeed.

This means we can write

$$X = Z_{\Pi} \cup Attr_{\Pi}^{\mathbb{B}_X}(N).$$

We are now almost ready to define the winning strategy for Π which keeps the token inside X. Recall that X is a $\overline{\Pi}$-trap, so that for each $b \in X \cap B_{\Pi}$, we may pick an arbitrary element

$k(b) \in E[b] \cap X$. Now define the following strategy h in $\mathcal{G}$ for Π on X.

$$h(b) := \begin{cases} k(b) & \text{if } b \in N \\ attr_\Pi(N)(b) & \text{if } b \in Attr_\Pi^{\mathbb{B}_X}(N) \setminus N \\ f_\Pi(b) & \text{if } b \in Z_\Pi = Z. \end{cases}$$

It is left as an exercise for the reader to show that h is indeed a winning strategy for Π in $\mathcal{G}$ and that it keeps the pebble in X. QED

Finally, the assertion made in Theorem 5.22 follows directly from this proposition because by definition of paradises there now exists for every point $b \in B$ a positional winning strategy for the game $\mathcal{G}(\mathbb{B}, \Omega)$.

- ▶ `strategies as 1-player games`
- ▶ `automatic moves`

5.5 Size issues and algorithmic aspects

5.6 Game equivalences

Notes

The application of game-theoretic methods in the area of logic and automata theory goes back to work of Büchi. The positional determinacy of parity games was proved independently by Emerson & Jutla [11] and by Mostowski in an unpublished technical report. Our proof of this result is based on Zielonka [29].

Exercises

Exercise 5.1 (positional determinacy of reachability games) Give a direct proof of the positional determinacy of reachability games, that is: prove Theorem 5.17.

Exercise 5.2 (regular games & finite memory strategies) An infinite game $\mathcal{G} = \langle B_0, B_1, E, W\rangle$ is called *regular* if there exists an ω-regular language L over some finite alphabet C and a colouring $\Gamma : B \to C$, such that player 0 wins $(p_i)_{i<\omega} \in B^\omega$ precisely if the induced sequence $(\Gamma(p_i))_{i<\omega} \in C^\omega$ belongs to L.

A strategy α for player Π in an infinite game $\mathcal{G} = \langle B_0, B_1, E, W\rangle$ is a *finite memory strategy* if there exists a finite set M, called the *memory set*, an element $m_I \in M$ and a map $(\alpha_1, \alpha_2) : B \times M \to B \times M$ such that for all pairs of sequences $p_0 \cdots p_k \in B^*$ and $m_0 \cdots m_k \in M^*$: if $m_0 = m_I$, $p_0 \cdots p_k \in \mathrm{PM}_\Pi$ and $m_{i+1} = \alpha_2(p_i, m_i)$ (for all $i < k$), then $\alpha(p_0 \cdots p_k) = \alpha_1(p_k, m_k)$.

Now let $\mathcal{G}$ be a regular game.

(a)* Show that $\mathcal{G}$ is determined, and that player 0 has a finite memory strategy which is winning for her in $\mathcal{G}@p$ for every $p \in \mathrm{Win}_0$.
Hint: define an auxiliary game with positions $B \times M$, where M is the carrier of a deterministic parity automaton $\mathbb{M}$ recognizing L.

(b) Does the same statement hold for player 1? That is, if $p \in \mathrm{Win}_1$, can you now conclude that player 1 has a winning finite memory strategy in $\mathcal{G}@p$?

6 Parity formulas & model checking

In this chapter we introduce *parity formulas*. In short, these are graph-based modal formulas with an added parity condition, that will allow us to view the evaluation games of μ-calculus formulas as instances of parity games. Providing a link between the world of μ-calculus formulas and that of parity games, they illuminate the complexity-theoretic analysis of the model checking problem of the modal μ-calculus. Parity formulas can also be studied in their own right, as an interesting generalisation of the regular (tree-based) μ-calculus formulas.

6.1 Parity formulas

We start with the basic definition of a parity formula. Recall that, given a set P of proposition letters, we define the sets $\mathtt{Lit}(\mathsf{P})$ and $\mathtt{At}(\mathsf{P})$ of *literals* and *atomic formulas* over P by setting $\mathtt{Lit}(\mathsf{P}) := \{p, \overline{p} \mid p \in \mathsf{P}\}$ and $\mathtt{At}(\mathsf{P}) := \mathtt{Lit}(\mathsf{P}) \cup \{\top, \bot\}$, respectively.

Definition 6.1 Let P be a finite set of proposition letters. A *parity formula over* P is a quintuple $\mathbb{G} = (V, E, L, \Omega, v_I)$, where

a) (V, E) is a finite, directed graph, with $|E[v]| \leq 2$ for every vertex v;
b) $L : V \to \mathtt{At}(\mathsf{P}) \cup \{\wedge, \vee, \Diamond, \Box, \epsilon\}$ is a labelling function;
c) $\Omega : V \stackrel{\circ}{\to} \omega$ is a partial map, the *priority* map of $\mathbb{G}$; and
d) v_I is a vertex in V, referred to as the *initial* node of $\mathbb{G}$;

such that

1) $|E[v]| = 0$ if $L(v) \in \mathtt{At}(\mathsf{P})$, and $|E[v]| = 1$ if $L(v) \in \{\Diamond, \Box\} \cup \{\epsilon\}$;
2) every cycle of (V, E) contains at least one node in $\mathsf{Dom}(\Omega)$.

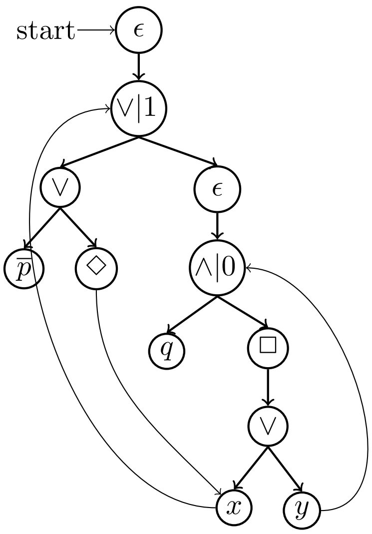
A node $v \in V$ is called *silent* if $L(v) = \epsilon$, *constant* if $L(v) \in \{\top, \bot\}$, *literal* if $L(v) \in \mathtt{Lit}(\mathsf{P})$, *atomic* if it is either constant or literal, *boolean* if $L(v) \in \{\wedge, \vee\}$, and *modal* if $L(v) \in \{\Diamond, \Box\}$. Elements of $\mathsf{Dom}(\Omega)$ will be called *states*. We say that a proposition letter q *occurs* in $\mathbb{G}$ if $L(v) \in \{q, \overline{q}\}$ for some $v \in V$. $\lhd$

Example 6.2 In Figure 2 we give two examples of parity formulas. The picture on the left displays a parity formula that is directly based on the μ-calculus formula $\xi = \mu x.(\overline{p} \vee \Diamond x) \vee \nu y.(q \wedge \Box(x \vee y))$, by adding *back edges* to the subformula dag of ξ. The picture on the right displays a parity formula that is based on a rather more entangled graph. $\lhd$

The definition of parity formulas needs little explanation. Condition 2) says that every cycle must pass through at least one state; this is needed to provide a winner for infinite matches of the evaluation games that we use to define the semantics of parity formulas. The rules (admissible moves) in this evaluation game are completely obvious.

Definition 6.3 Let $\mathbb{S} = (S, R, V)$ be a Kripke model for a set P of proposition letters, and let $\mathbb{G} = (V, E, L, \Omega, v_I)$ be a parity P-formula. The *evaluation game* $\mathcal{E}(\mathbb{G}, \mathbb{S})$ is the parity game (G, E, Ω') of which the board consists of the set $V \times S$, the priority map $\Omega' : V \times S \to \omega$ is given by

$$\Omega'(v, s) := \begin{cases} \Omega(v) & \text{if } v \in \mathsf{Dom}(\Omega) \\ 0 & \text{otherwise,} \end{cases}$$



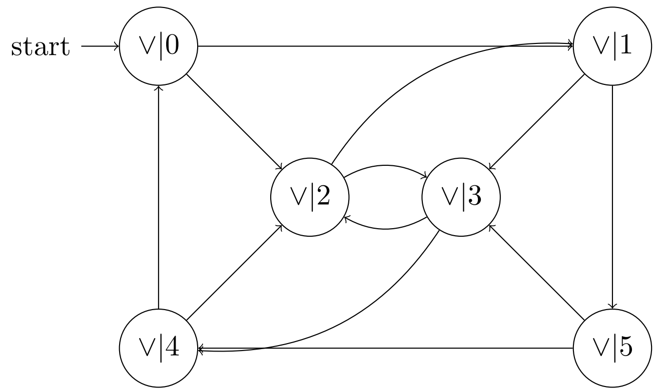


Figure 2: Two parity formulas

and the game graph is given in Table 8. Note that we do not need to assign a player to positions that admit a single move only. $\lhd$

Definition 6.4 We say that a parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$ *holds at* or *is satisfied by* a pointed Kripke model $(\mathbb{S}, s)$, notation: $\mathbb{S}, s \Vdash \mathbb{G}$, if the pair (v_I, s) is a winning position for $\exists$ in $\mathcal{E}(\mathbb{G}, \mathbb{S})$. We let $\mathcal{Q}(\mathbb{G})$ denote the *query* of $\mathbb{G}$, that is, the class of pointed Kripke models where $\mathbb{G}$ holds, and we call two parity formulas $\mathbb{G}$ and $\mathbb{G}'$ *equivalent* if they determine the same query, notation: $\mathbb{G} \equiv \mathbb{G}'$. We will use the same terminology and notation to compare parity formulas with standard formulas. $\lhd$

The two key complexity measures of a parity formula, viz., size and index, both have perspicuous definitions. We will introduce these measures here, together some other useful notions pertaining to parity formulas.

Definition 6.5 The *size* of a parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$ is defined as its number of nodes: $|\mathbb{G}| := |V|$. $\lhd$

Next to size, as the second fundamental complexity measure for a parity formula we need is its *index*, which corresponds to the alternation depth of regular formulas. It concerns the degree of alternation between odd and even positions in an infinite match of the evaluation game, and it is thus closely related to the range of the priority map of the formula. The most straightforward approach would be to define the index of a parity formula as the size of this range; a slightly more sophisticated approach is a *clusterwise* version of this.

Definition 6.6 Let $\mathbb{G} = (V, E, L, \Omega, v_I)$ be a parity formula, and let u and v be vertices in V. We say that v is *active* in u if E^+uv, and we let $\bowtie_E \subseteq V \times V$ hold between u and v

Position	Player	Admissible moves
(v, s) with $L(v) = p$ and $s \in V(p)$	$\forall$	$\varnothing$
(v, s) with $L(v) = p$ and $s \notin V(p)$	$\exists$	$\varnothing$
(v, s) with $L(v) = \overline{p}$ and $s \in V(p)$	$\exists$	$\varnothing$
(v, s) with $L(v) = \overline{p}$ and $s \notin V(p)$	$\forall$	$\varnothing$
(v, s) with $L(v) = \bot$	$\exists$	$\varnothing$
(v, s) with $L(v) = \top$	$\forall$	$\varnothing$
(v, s) with $L(v) = \epsilon$	-	$E[v] \times \{s\}$
(v, s) with $L(v) = \vee$	$\exists$	$E[v] \times \{s\}$
(v, s) with $L(v) = \wedge$	$\forall$	$E[v] \times \{s\}$
(v, s) with $L(v) = \Diamond$	$\exists$	$E[v] \times R[s]$
(v, s) with $L(v) = \Box$	$\forall$	$E[v] \times R[s]$

Table 8: The evaluation game $\mathcal{E}(\mathbb{G}, \mathbb{S})$

is u is active in v and vice versa, i.e., $\bowtie_E := E^+ \cap (E^{-1})^+$. We let $\equiv_E$ be the equivalence relation generated by $\bowtie_E$; the equivalence classes of $\equiv_E$ will be called *clusters*. A cluster C is called *degenerate* if it is a singleton $\{v\}$ such that v is not active in itself, and *nondegenerate* otherwise.

The collection of clusters of a parity formula $\mathbb{G}$ is denoted as $Clus(\mathbb{G})$, and we say that a cluster C is *higher* than another cluster C' if for every $u \in C$ there is a $u \in C'$ such that E^+uu'. $\lhd$

Note that in a nondegenerate cluster there is a nontrivial path between any pair of vertices, and observe that the 'higher than' relation between clusters is a strict partial order.

Intuitively, vertices belong to the same (nondegenerate) cluster if they can jointly occur infinitely often in some infinite match of some acceptance game for the formula.

Proposition 6.7 *Let $\tau = (t_n)_{n\in\omega}$ be an infinite path through the graph of a parity formula $\mathbb{G}$. Then $\mathbb{G}$ has a unique cluster C such that, for some k, all t_n with $n > k$ belong to C.*

As a corollary of this, the relative priorities of states only matter if we stay in the same cluster. We will define the index of a parity formula in terms of the maximal length of so-called alternating Ω-chains, where we will only consider chains of states that belong to the same cluster.

Definition 6.8 Let $\mathbb{G} = (V, E, L, \Omega, v_I)$ be a parity formula. An *alternating Ω-chain* of *length k* in $\mathbb{G}$ is a finite sequence $v_1 \cdots v_k$ of states that all belong to the same cluster, and satisfy, for all $i < k$, that $\Omega(v_i) < \Omega(v_{i+1})$ while v_i and v_{i+1} have different parity. Such a chain is called an η-chain if $\Omega(v_k)$ has parity η (where we recall that we associate even numbers with ν and odd numbers with μ). $\lhd$

Note that a parity formula $\mathbb{G}$ has alternating chains iff it has states, i.e., $\mathsf{Dom}(\Omega) \neq \varnothing$.

Definition 6.9 The *index* of a parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$ is defined as the maximal length of an alternating Ω-chain in $\mathbb{G}$. As a special case we put $ind(\mathbb{G}) = 0$ if $\mathbb{G}$ has no alternating Ω-chains. $\triangleleft$

Observe that if $\mathbb{G}$ is cycle-free then we can assume that the range of Ω is empty. Thus, every cycle-free parity formula is equivalent to one with index zero.

A useful consequence of the above definition is that parity formulas that are *parity variants* will have the same index.

Definition 6.10 A *parity variant* of a parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$ is a parity formula $\mathbb{G} = (V, E, L, \Omega', v_I)$ such that (i) $\Omega(v) \equiv_2 \Omega'(v)$, for all v, and (ii) $\Omega(u) < \Omega(v)$ iff $\Omega'(u) < \Omega'(v)$, for all u and v that belong to the same cluster but have different parity. $\triangleleft$

It is easy to see that parity variants are semantically equivalent, and have the same index. From this it follows that there are certain normal forms for parity formulas.

Definition 6.11 A parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$ is called *lean* if Ω is injective, and *tight* if for any cluster C, the range of Ω on C is connected, that is, of the form $\mathsf{Ran}(\Omega{\upharpoonright_C}) = [k, \ldots, n]$ for some natural numbers k, n with $k \leq n$. Here we define $[k, \ldots, n] := \{i \in \omega \mid k \leq i \leq n\}$. $\triangleleft$

It is not hard to see that every parity formula can be effectively transformed into either a lean or a tight parity variant; for the tight case, see Proposition 6.17 below. Furthermore, it is rather obvious that for a tight parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$, we have

$$ind_{\mathbb{G}}(C) = |\mathsf{Ran}(\Omega{\upharpoonright_C})|, \tag{41}$$

so that for these devices our definition of index matches the one we mentioned earlier, viz., in terms of the clusterwise size of the range of the priority map.

6.2 Basics

Priority maps and parity preorders

Quite often the priority function of a parity formula is induced by some kind of (clusterwise) preorder on its sets of states. It will be convenient to introduce some terminology.

Definition 6.12 A *parity preorder* is a structure $\mathbb{P} = (P, \sqsubseteq, p)$, where $(P, \sqsubseteq)$ is a directed preorder and $p : P \to \{0, 1\}$ is a map such that $u \equiv v$ implies $p(u) = p(v)$. Here we let $\equiv$ denote the equivalence relation induced by $\sqsubseteq$. $\triangleleft$

To make a proper link with parity formulas, note that the preorders we have in mind here are based on the states in a single *cluster* of a parity formula. Thus, for instance, the relation $\equiv$ is not the relation $\equiv_E$ of Definition 6.6.

Observe that by directedness, every parity preorder has an $\equiv$-cell of $\sqsubseteq$-*maximal* elements, and that these points all have the same parity.

Definition 6.13 Fix a parity preorder $\mathbb{P} = (P, \sqsubseteq, p)$. An *alternating chain* in $\mathbb{P}$ of *length* k in $\mathbb{P}$ is a finite sequence $v_1 \cdots v_k$ of states such that, for all $i < k$, $v_i \sqsubseteq v_{i+1}$ while v_i and v_{i+1} have different parity. Given a point $v \in P$ we define $h^{\uparrow}(v)$ (respectively, $h^{\downarrow}(v)$) as the maximal length of an alternating chain starting at v (ending at v, respectively), and we let $ad(\mathbb{P})$ denote the *alternation depth* of $\mathbb{P}$, i.e., the maximal length of an alternating chain in $\mathbb{P}$.

We define the following map $\Omega_{\mathbb{P}} : P \to \omega$:

$$\Omega_{\mathbb{P}}(v) := \begin{cases} ad(\mathbb{P}) - h^{\uparrow}(v) & \text{if } ad(\mathbb{P}) - h^{\uparrow}(v) \equiv_2 p(v) \\ ad(\mathbb{P}) - h^{\uparrow}(v) + 1 & \text{if } ad(\mathbb{P}) - h^{\uparrow}(v) \not\equiv_2 p(v), \end{cases} \tag{42}$$

and we will call this map the *priority map induced by* $\mathbb{P}$. $\lhd$

Intuitively, we define $\Omega_{\mathbb{P}}(v)$ to be $ad(\mathbb{P}) - h^{\uparrow}(v)$, possibly with a corrective '+ 1' to ensure the right parity. As a fairly direct consequence of this definition, it follows that $u \sqsubseteq v$ implies $\Omega(u) \leq \Omega(v)$, with an inequality holding if u and v have different parity. In particular, all $\sqsubseteq$-maximal points obtain the same priority which is the maximal Ω-value reached. More information about the construction is provided by the next proposition.

Proposition 6.14 *Let $\mathbb{P} = (P, \sqsubseteq, p)$ be a parity preorder, and let Ω be its induced priority map. Then for every $u, v \in P$, it holds that $\Omega(v) \equiv_2 p(v)$, that $u \sqsubseteq v$ implies $\Omega(u) \leq \Omega(v)$, and that $u \sqsubset v$ and $p(u) \neq p(v)$ implies $\Omega(u) < \Omega(v)$, Furthermore,* $\mathsf{Ran}(\Omega)$ *is connected, and*

$$|\mathsf{Ran}(\Omega)| = ad(\mathbb{P}). \tag{43}$$

Proof. We leave it for the reader to verify that, for every $u, v \in P$, we have that $\Omega(v) \equiv_2 p(v)$, that $u \sqsubseteq v$ implies $\Omega(u) \leq \Omega(v)$, that $u \sqsubset v$ implies $\Omega(u) < \Omega(v)$, and that $\mathsf{Ran}(\Omega)$ is connected. For a proof of (43) the reader is invited to check that $\mathsf{Ran}(\Omega)$ equals either $[0, \ldots, H-1]$ or $[1, \ldots, H]$, depending on the parity of H and the parity value $p(m)$ for any $\sqsubseteq$-maximal point m:

	$p(m)$ even	$p(m)$ odd
H even	$[1, \ldots, H]$	$[0, \ldots, H-1]$
H odd	$[0, \ldots, H-1]$	$[1, \ldots, H]$

From this (43) is immediate. QED

Remark 6.15 To see how parity formulas may be defined on the basis of parity preorders, let (V, E, L) be a directed graph with a labelling L as in a parity formula. Furthermore, let p be a partial map from V to $\{0, 1\}$, and let $\sqsubseteq$ be a preorder on $\mathsf{Dom}(p)$ such that, for every cluster C of (V, E), the structure $(C \cap \mathsf{Dom}(p), \sqsubseteq, p)$ is a parity preorder. Finally, assume that on every E-cycle there is a state (i.e., an element of the domain of p) of maximal priority; that is, there is a state u on the cycle such that $v \sqsubseteq u$ for every state v on the same cycle. Then we may associate a (clusterwise defined) priority map Ω on V such that any infinite path $\pi = (v_n)_{n \in \omega}$ through the graph meets the parity condition for Ω iff there is a state $u \in \mathsf{Dom}(p) \cap \mathit{Inf}(\pi)$ such that $v \sqsubseteq u$ for every state $v \in \mathsf{Dom}(p) \cap \mathit{Inf}(\pi)$. $\lhd$

Remark 6.16 A simpler and possibly more natural definition would be to set

$$\Omega_{\mathbb{P}}(v) := \begin{cases} h^{\downarrow}(v) & \text{if } h^{\downarrow}(v) \equiv_2 p(v) \\ h^{\downarrow}(v) + 1 & \text{if } h^{\downarrow}(v) \not\equiv_2 p(v), \end{cases} \tag{44}$$

but (42) gives a slightly better link with the priority map.

To see this, consider the parity preorder $\mathbb{P}$ based on the three element set $\{u, v, w\}$ which we partially order by putting $u \sqsubseteq w$ and $v \sqsubseteq w$ (while not making any link between u and v). If we put $p(u) = p(w) = 0$ and $p(v) = 1$, then it is easily verified that $ad(\mathbb{P}) = 2$. Were we now to define $\Omega'(x)$ as in (44), we would get $\Omega'(u) = 0$, $\Omega'(v) = 1$ and $\Omega'(w) = 2$, implying that $|\mathsf{Ran}(\Omega)| = 3$. However, defining $\Omega_{\mathbb{P}}$ as in (42), we obtain that $\Omega_{\mathbb{P}}(u) = \Omega_{\mathbb{P}}(w) = 2$, while $\Omega_{\mathbb{P}}(v) = 1$, so that we find $|\mathsf{Ran}(\Omega_{\mathbb{P}})| = ad(\mathbb{P})$ indeed. $\lhd$

As a first application of this Proposition, we show that every parity formula is equivalent to a tight one. The proof of Proposition 6.17 is left for the reader.

Proposition 6.17 *For every parity formula $\mathbb{G}$ there is a tight parity formula $\mathbb{G}'$ such that $\mathbb{G}' \equiv \mathbb{G}$ and $ind(\mathbb{G}') \leq ind(\mathbb{G})$.*

Operations on parity formulas

Parity formulas are interesting logical objects in their own right, and so one might want to develop their theory. To start with, it is fairly easy to define various operations on parity formulas, such as modal and boolean operations (including negation), least and fixpoint operations, and substitution.

▶ `Examples to follow.`

Morphisms between parity formulas

Furthermore, it would be of interest to study various *structural* notions of equivalence between parity formulas.

▶ `More to follow.`

6.3 From regular formulas to parity formulas

Since the evaluation game for parity formulas is given as a parity game, we immediately get a quasi-polynomial upper bound on the time complexity of the *model checking* problem for parity formulas. Recall that the size of a (pointed) labelled transition system is simply defined as the number of points in the model.

Definition 6.18 The *model checking problem* for parity formulas is the problem to compute whether $\mathbb{S}, s \Vdash \mathbb{G}$, where $\mathbb{S}$ is a (finite) labelled transition system, and $\mathbb{G}$ is a parity formula. $\lhd$

Theorem 6.19 *The model checking problem for parity formulas can be solved in time $2^{(\log(mn))(\log(k))}$, where m is the size of the labelled transition system, and n and k are the size and index of the parity formula, respectively.*

So how can we use this result to analyse the computational complexity of the model checking problem for regular formulas (i.e., formulas of the modal μ-calculus? The key idea, and the topic of this section, will be to transform a μ-calculus formula into an equivalent parity formula *of minimal size and index*. It should come as no surprise that the index of this parity formula will somehow correspond to the alternation depth of the formula, while the size of the parity formula will clearly depend on the graph structure that we pick to represent the original formula.

Basically, there are *three* natural candidates for such a graph: next to the syntax tree, these are the *subformula dag* and the *closure graph* of the formula. Note that each of these three structures induces a natural *size measure* of μ-calculus formulas, respectively length, dag-size, and (closure-)size. Since we will not be interested much in working with length as a size measure, this means that in the following two subsections we will focus on the latter two graph structures.

Recall that the *subformula dag* of a clean formula ξ is the pointed graph $\mathbb{D}_\xi := (\mathit{Sfor}(\xi), \triangleright_0, \xi)$, where $\triangleright_0$ is the converse of the direct subformula relation $\triangleleft_0$. The *closure graph* of a tidy formula $\xi \in \mu\mathtt{ML}$ is the structure $\mathbb{C}_\xi = (\mathit{Clos}(\xi), \to_C, \xi)$, where $\to_C$ is the trace relation (restricted to the closure of ξ).

6.3.1 Parity formulas on the subformula dag

The following theorem shows that for a clean formula, we can indeed obtain an equivalent parity formula which is based on its *subformula dag*.

Theorem 6.20 *There is an algorithm that constructs, for a clean formula $\xi \in \mu\mathtt{ML}(\mathsf{P})$, an equivalent parity formula $\mathbb{H}_\xi$ over P, such that $|\mathbb{H}_\xi| = |\xi|^d$ and $ind(\mathbb{H}_\xi) = ad(\xi)$.*

The basic idea underlying the proof of Theorem 6.20 is to view the evaluation games for clean formulas in $\mu\mathtt{ML}$ as instances of parity games. Given an arbitrary formula $\xi \in \mu\mathtt{ML}$, we then need to see which modifications are needed to turn the subformula dag $(\mathit{Sfor}(\xi), \triangleright_0)$ a parity formula $\mathbb{H}_\xi$ such that, for any model $\mathbb{S}$, the evaluation games $\mathcal{E}(\xi, \mathbb{S})$ and $\mathcal{E}(\mathbb{H}_\xi, \mathbb{S})$ are more or less identical. Clearly, the fact that the *positions* of the evaluation game $\mathcal{E}(\xi, \mathbb{S})$ are given as the pairs in the set $\mathit{Sfor}(\xi) \times S$, means that we can take the set

$$V_\xi := \mathit{Sfor}(\xi)$$

as the carrier of $\mathbb{H}_\xi$ indeed.

Looking at the admissible moves in the two games, it turns out that we cannot just take the converse direct subformula relation $\triangleright_0$ as the edge relation of $\mathbb{H}_\xi$: we need to add all *back edges* from the set

$$B_\xi := \{(x, \delta_x) \mid x \in BV(\xi)\},$$

where, as usual, we let δ_x denote the unique formula such that, for some $\eta \in \{\mu, \nu\}$ the formula $\eta x.\delta_x$ is a subformula of ξ. In fact, if we write D_ξ for the relation $\triangleright_0$, restricted to $Sfor(\xi)$, then we can take

$$E_\xi := D_\xi \cup B_\xi,$$

as the edge relation of $\mathbb{H}_\xi$. Furthermore, the labelling map L_ξ is naturally defined via the following case distinction:

$$L_\xi(\varphi) := \begin{cases} \varphi & \text{if } \varphi \in \{\top, \bot\} \cup \{p, \overline{p} \mid p \in FV(\xi)\} \\ \odot & \text{if } \varphi \text{ is of the form } \varphi_0 \odot \varphi_1 \text{ with } \odot \in \{\wedge, \vee\} \\ \heartsuit & \text{if } \varphi \text{ is of the form } \heartsuit\psi \text{ with } \heartsuit \in \{\Diamond, \Box\} \\ \epsilon & \text{if } \varphi \text{ is of the form } \eta_x x.\delta_x \text{ with } \eta \in \{\mu, \nu\} \\ \epsilon & \text{if } \varphi \in BV(\xi). \end{cases}$$

With this definition, it is easy to see that the *boards* of the two evaluation games $\mathcal{E}(\xi, \mathbb{S})$ and $\mathcal{E}(\mathbb{H}_\xi, \mathbb{S})$ are *isomorphic*, for any labeled transition system $\mathbb{S}$. As the initial node v_ξ of $\mathbb{H}_\xi$ we simply take

$$v_\xi := \xi.$$

In order to finish the definition of the parity formula $\mathbb{H}_\xi$ it is then left to come up with a suitable priority map Ω_ξ on V_ξ. Since the winning conditions of the evaluation game for the formula ξ are defined in terms of the priority ordering $\leq_\xi$ on the collection $BV(\xi)$ of bound variables of ξ, it seems natural to take these bound variables of ξ as the states of $\mathbb{H}_\xi$, that is, the nodes for which a priority is defined. It will be more convenient, however, to take the *unfoldings* of these bound variables instead; that is, we will take $\mathsf{Dom}(\Omega_\xi) = \{\delta_x \mid x \in BV(\xi)\}$.

Now if we are only interested in the *equivalence* of ξ and $\mathbb{H}_\xi$, any priority map Ω will be fine, as long as it satisfies two conditions: (i) $\Omega(\delta_x) \leq \Omega(\delta_y)$ iff $x \leq_\xi y$, and (ii) $\Omega(\delta_x)$ is even iff x is a ν-variable. For instance, a straightforward suggestion would be the following. Given $x \in BV(\xi)$, let $h^\downarrow(x)$ be the maximal length of an alternating fixpoint chain ending at x, and set

$$\Omega'(\delta_x) := \begin{cases} h^\downarrow(x) & \text{if } h^\downarrow(x) \text{ has the same parity as } \eta_x \\ h^\downarrow(x) + 1 & \text{otherwise.} \end{cases}$$

where we recall that μ and ν have, respectively, odd and even parity. It is easy to verify that with this definition Ω satisfies the conditions (i) and (ii), so that we find that $\xi \equiv (V_\xi, E_\xi, L_\xi, \Omega', v_\xi)$ indeed.

In order to get an *exact* match of the index of $\mathbb{H}_\xi$ and the alternation depth of ξ we need to work a bit harder, cf. the discussion in Remark 6.16 about Definition 6.13.

Definition 6.21 Given a bound variable $x \in BV(\xi)$, let $h^\uparrow_\xi(x)$ be the maximal length of an alternating $<_\xi$-chain of fixpoint variables *starting at* x. Furthermore, let $h_\xi(x)$ be the maximal length of an alternating $<_\xi$-chain in the cluster of x. Then we define

$$\Omega_\xi(\delta_x) := \begin{cases} h_\xi(x) - h^\uparrow_\xi(x) & \text{if } h_\xi(x) - h^\uparrow_\xi(x) \text{ has the same parity as } \eta_x \\ h_\xi(x) - h^\uparrow_\xi(x) + 1 & \text{otherwise.} \end{cases}$$

Finally, we define $\mathbb{H}_\xi := (V_\xi, E_\xi, L_\xi, \Omega_\xi, v_\xi)$, where V_ξ, E_ξ, L_ξ, and v_ξ are as defined above. $\triangleleft$

Proof of Theorem 6.20. In the light of the above discussion, the equivalence of ξ and $\mathbb{H}_\xi$ follows from the easily verified fact that Ω_ξ satisfies the conditions (i) and (ii) mentioned above. It is immediate by the definitions that $|\mathbb{H}_\xi| = |Sfor(\xi)| = |\xi|^d$. Finally, we obtain $ind(\mathbb{H}_\xi) = ad(\xi)$ as a consequence of the Propositions 2.51 and 6.14. QED

6.3.2 Parity formulas on the closure graph

The next theorem states that for an arbitrary tidy formula, we can find an equivalent parity formula that is based on the formula's *closure graph*, and has an index which is bounded by the alternation depth of the formula.

Theorem 6.22 *There is a construction transforming an arbitrary tidy formula $\xi \in \mu\mathtt{ML}$ into an equivalent parity formula $\mathbb{G}_\xi$ which is based on the closure graph of ξ, so that $|\mathbb{G}| = |\xi|$; in addition we have $ind(\mathbb{G}_\xi) \leq ad(\xi)$.*

When it comes to complexity issues, this is in fact the main result that bridges the gap between the world of formulas and that of automata and parity games. In particular, as an immediate corollary of Theorem 6.22 and the quasi-polynomial time complexity result on the model checking problem for parity formulas (Theorem 6.19), we find that model checking for μ-calculus formulas can be solved in quasi-polynomial time.

Theorem 6.23 *The model checking problem for μ-calculus formulas can be solved in time $2^{(\log(mn))(\log(k)}$, where n is the size of the formula, k is its alternation depth, and m is the size of the labelled transition system of size m.*

The priority map that we will define on the closure graph of a tidy formula is in fact *global* in the sense that it can be defined uniformly for all (tidy) formulas, independently of any ambient formula. Furthermore, we will base this map on a partial order of fixpoint formulas, the *closure priority relation* $\sqsubseteq_C$ that we will introduce now. Recall that $\trianglelefteq_f$ denotes the *free* subformula relation introduced in Definition 2.39.

Definition 6.24 We let $\equiv_C$ denote the equivalence relation generated by the relation $\to_C$, in the sense that: $\varphi \equiv_C \psi$ if $\varphi \twoheadrightarrow_C \psi$ and $\psi \twoheadrightarrow_C \varphi$. We will refer to the equivalence classes of $\equiv_C$ as *(closure) clusters*, and denote the cluster of a formula φ as $C(\varphi)$.

We define the *closure priority relation* $\sqsubseteq_C$ on fixpoint formulas by putting $\varphi \sqsubseteq_C \psi$ precisely if $\psi \twoheadrightarrow_C^\psi \varphi$, where $\twoheadrightarrow_C^\psi$ is the relation given by $\rho \twoheadrightarrow_C^\psi \sigma$ if there is a trace $\rho = \chi_0 \to_C \chi_1 \to_C \cdots \to_C \chi_n = \sigma$ such that $\psi \trianglelefteq_f \chi_i$, for every $i \in [0, \ldots, n]$. We write $\varphi \sqsubset_C \psi$ if $\varphi \sqsubseteq_C \psi$ and $\psi \not\sqsubseteq_C \varphi$. $\lhd$

The above definition of the closure priority relation is rather involved, but this seems to be unavoidable if we want to have an exact match of the index of $\mathbb{G}_\xi$ to the alternation depths of ξ. A simpler alternative (which does not give such an exact match) is given in Remark 6.31 below.

To avoid confusion let us mention right away here that $\equiv_C$ is not necessarily the equivalence relation induced by $\sqsubset_C$: For starters, $\sqsubset_C$ is only defined on fixpoint formulas, while $\equiv_C$ relates *all* formulas in a cluster. Here are some further basic observations on the relations $\equiv_C$ and $\sqsubseteq_C$.

Proposition 6.25 *1) The relation $\sqsubseteq_C$ is a partial order.*

2) The relation $\sqsubseteq_C$ is included in the closure equivalence relation: $\varphi \sqsubseteq_C \psi$ implies $\varphi \equiv_C \psi$.

3) The relation $\sqsubseteq_C$ is included in the converse free subformula relation: $\varphi \sqsubseteq_C \psi$ implies $\psi \trianglelefteq_f \varphi$.

Proof. For item 1) we need to show that $\sqsubseteq_C$ is reflexive, transitive and antisymmetric. Reflexivity is obvious, and antisymmetry follows from 3). For transitivity assume that $\varphi \sqsubseteq_C \psi$ and $\psi \sqsubseteq_C \chi$ hold. By definition this means that $\psi \twoheadrightarrow_C^{\psi} \varphi$ and $\chi \twoheadrightarrow_C^{\chi} \psi$. The latter entails that $\chi \trianglelefteq_f \psi$ and the former means that there is some $\rightarrow_C$-trace from ψ to φ such that ψ is a free subformula of every formula along this trace. Because $\chi \trianglelefteq_f \psi$ and $\trianglelefteq_f$ is transitive it then also holds that χ is a free subformula of every formula on the trace from ψ to φ. Composing this trace with the one from χ to ψ we obtain a trace from χ to φ such that χ is a free subformula of all formulas along this trace. Hence $\chi \twoheadrightarrow_C^{\chi} \varphi$ and so $\varphi \sqsubseteq_C \chi$.

For item 2) we assume that $\varphi \sqsubseteq_C \psi$ and need to show that $\varphi \twoheadrightarrow_C \psi$ and $\psi \twoheadrightarrow_C \varphi$. The assumption $\varphi \sqsubseteq_C \psi$ means that $\psi \twoheadrightarrow_C^{\psi} \varphi$ which clearly entails $\psi \twoheadrightarrow_C \varphi$. But, as already observed above, $\psi \twoheadrightarrow_C^{\psi} \varphi$ also entails that $\psi \trianglelefteq_f \varphi$, from which $\varphi \twoheadrightarrow_C \psi$ follows by Proposition 2.40.

Item 3) is immediate by the definition of $\sqsubseteq_C$. QED

Since $\sqsubseteq_C$ is a partial order (and hence a preorder), we may use Definition 6.13 to base a priority map on it. The details are spelled out below.

Definition 6.26 An *alternating $\sqsubset_C$-chain* of length n is a sequence $(\eta_i x_i.\chi_i)_{i\in[1,\dots,n]}$ of tidy fixpoint formulas such that $\eta_i x_i.\chi_i \sqsubset_C \eta_{i+1} x_{i+1}.\chi_{i+1}$ and $\eta_{i+1} = \overline{\eta_i}$ for all $i \in [0,\dots,n-1]$. We say that such a chain *starts at* $\eta_1 x_1.\chi_1$ and *leads up to* $\eta_n x_n.\chi_n$.

Given a tidy fixpoint formula ξ, we let $h^{\uparrow}(\xi)$ and $h^{\downarrow}(\xi)$ denote the maximal length of any alternating $\sqsubset_C$-chain starting at, respectively leading up to, ξ. Given a closure cluster C, we let $cd(C)$ denote the *closure depth* of C, i.e., the maximal length of any alternating $\sqsubset_C$-chain in C.

The *global priority function* $\Omega_g : \mu\mathtt{ML}^t \to \omega$ is defined cluster-wise, as follows. Take an arbitrary tidy fixpoint formula $\psi = \eta y.\varphi$, and define

$$\Omega_g(\psi) := \begin{cases} cd(C(\psi)) - h^{\uparrow}(\psi) & \text{if } cd(C(\psi)) - h^{\uparrow}(\psi) \text{ has parity } \eta \\ cd(C(\psi)) - h^{\uparrow}(\psi) + 1 & \text{if } cd(C(\psi)) - h^{\uparrow}(\psi) \text{ has parity } \overline{\eta}. \end{cases} \quad (45)$$

Here we recall that we associate μ and ν with odd and even parity, respectively.

If ψ is not of the form $\eta y.\varphi$, we leave $\Omega_g(\psi)$ undefined. ◁

We are now ready for the definition of the parity formula $\mathbb{G}_\xi$ corresponding to a tidy formula ξ.

Definition 6.27 Fix some tidy formula ξ. We define $\mathbb{C}_\xi$ be the closure graph $(Clos(\xi), \rightarrow_C)$ of ξ, expanded with the natural labelling L_C given by

$$L_C(\varphi) = \begin{cases} \varphi & \text{if } \varphi \in \mathtt{At}(\mathsf{P}) \\ \heartsuit & \text{if } \varphi = \heartsuit\psi \\ \odot & \text{if } \varphi = \psi_0 \odot \psi_1 \\ \epsilon & \text{if } \varphi = \eta x.\psi \end{cases}$$

Finally, we let $\mathbb{G}_\xi$ be the parity formula $\mathbb{G}_\xi := (\mathbb{C}_\xi, \Omega_g\restriction_{Clos(\xi)}, \xi)$. $\lhd$

The next Proposition gathers some facts about Ω_g, all of which are immediate consequences of Proposition 6.14. Recall that the *index* of a cluster in a parity formula is defined as the maximal length of an alternating chain in C, where alternation is expressed in terms of the priority map. With our definition of the global priority map Ω_g, the index of any cluster corresponds to the size of the range of Ω_g, restricted to the cluster.

Proposition 6.28 *1) Let $\xi = \eta x.\chi$ be a tidy fixpoint formula. Then $\Omega_g(\xi)$ has parity η.*

2) Let φ and ψ be tidy fixpoint formulas such that $\varphi \sqsubseteq_C \psi$. Then $\Omega_g(\varphi) \leq \Omega_g(\psi)$, and $\Omega_g(\varphi) < \Omega_g(\psi)$ if φ and ψ have different parity.

3) For any closure cluster C it holds that $cd(C) = ind(C) = |\mathsf{Ran}(\Omega_g\restriction_C)|$.

The following proposition shows that the global priority map indeed captures the right winner of infinite matches of the evaluation game.

Proposition 6.29 *1) For any finite trace $\rho_n \to_C \ldots \to_C \rho_1$ of tidy formulas there is a unique $\rho \in \{\rho_1, \ldots, \rho_n\}$ such that $\rho_n \twoheadrightarrow_C^\rho \rho_1$. Moreover, if ρ_1 is a fixpoint formula then so is ρ.*

2) For any infinite trace $\tau = (\xi_n)_{n\in\omega}$ of tidy formulas there is a unique fixpoint formula $\xi = \eta x.\chi$ which occurs infinitely often on τ and satisfies $\xi_n \sqsubseteq_C \xi$ for cofinitely many n. Here $\eta = \nu$ iff $\max\big(\{\Omega(\varphi) \mid \varphi$ occurs infinitely often on $\tau\}\big)$ is even.

Proof. In the proof we will use the following observation, the proof of which we leave as an exercise to the reader:

$$\begin{array}{l}\text{if } \xi \text{ is tidy and } \xi \to_C \psi \text{ then every } \varphi \trianglelefteqslant_f \psi \text{ satisfies either } \varphi \trianglelefteqslant_f \xi \text{ or } \xi \trianglelefteqslant_f \varphi, \\ \text{and in the latter case } \xi \text{ is a fixpoint formula.}\end{array} \tag{46}$$

We prove the proposition by induction over n, and note that we only need to worry about existence: if there would be a ρ and a ρ' meeting the constraints, we would find $\rho \trianglelefteqslant_f \rho'$ and $\rho' \trianglelefteqslant_f \rho$, implying $\rho = \rho'$.

The base case, where $\rho_n = \rho_1$, is trivial. For the induction step consider a trace $\rho_{n+1} \to_C \rho_n \to_C \ldots \to_C \rho_1$ and assume that the induction hypothesis holds for $\rho_n \to_C \ldots \to_C \rho_1$. Thus there is a ρ_i among $\rho_1, \ldots, \rho_n$ such that $\rho_n \twoheadrightarrow_C^{\rho_i} \rho_1$. We want to find a j such that $\rho_{n+1} \twoheadrightarrow_C^{\rho_j} \rho_j \twoheadrightarrow_C^{\rho_j} \rho_1$.

Because $\rho_{n+1} \to_C \rho_n$ we can use Proposition 46 to deduce that for every free subformula ψ' of ρ_n either $\psi' \trianglelefteqslant_f \rho_{n+1}$ or $\rho_{n+1} \trianglelefteqslant_f \psi'$. We have $\rho_i \trianglelefteqslant_f \rho_n$ since $\rho_n \twoheadrightarrow_C^{\rho_i} \rho_i$. Hence, we get either $\rho_i \trianglelefteqslant_f \rho_{n+1}$, in which case we can set $j := i$, or we get $\rho_{n+1} \trianglelefteqslant_f \rho_i$, in which case we can set $j := n+1$, because then $\rho_n \twoheadrightarrow_C^{\rho_i} \rho_i \twoheadrightarrow_C^{\rho_i} \rho_1$ implies $\rho_n \twoheadrightarrow_C^{\rho_j} \rho_i \twoheadrightarrow_C^{\rho_j} \rho_1$.

The 'moreover'-part is trivial in the base case. For the inductive step observe that we only reassign the ρ_j to ρ_{n+1} in the second case of the case distinction. But then Proposition 46 gives us that ρ_{n+1} must be a fixpoint formula.

Part 2) is more or less immediate by part 1) and the definitions. From this it follows by Proposition 6.14 that $\Omega_g(\varphi) \leq \Omega_g(\xi)$ for all φ that occur infinitely often on τ. Finally, that $\Omega_g(\xi)$ has the right parity was stated in Proposition 6.28. QED

As a fairly straightforward consequence of Proposition 6.29 and Proposition 2.38 we can prove the following result, which we shall need further on.

Proposition 6.30 *Every $\equiv_C$-cluster contains a unique fixpoint formula $\xi = \eta x.\chi$ such that $\xi \notin Clos(\chi)$. This formula is the $\sqsubseteq_C$-maximum element of its cluster.*

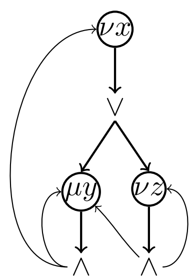
Remark 6.31 The definition of the priority map Ω_g and of the priority order $\sqsubset_C$ on which it is based, may look overly complicated. In fact, simpler definitions would suffice if we are only after the equivalence of a tidy formula with an associated parity formula that is based on its closure graph, i.e., if we do not need an exact match of index and alternation depth.

In particular, we could have introduced an alternative priority order $\sqsubset'_C$ by putting $\varphi \sqsubset'_C \psi$ if $\varphi \equiv_C \psi$ and $\psi \triangleleft_f \varphi$. If we would base a priority map Ω'_g on this priority order instead of on $\sqsubset_C$, then we could prove the equivalence of any tidy formula ξ with the associated parity formula $\mathbb{G}'_\xi := (\mathbb{C}_\xi, \Omega'_g\!\upharpoonright_{Clos(\xi)}, \xi)$. However, we would not be able to prove that the index of $\mathbb{G}'_\xi$ is bounded by the alternation depth of ξ.

To see this, consider the following formula:

$$\alpha_x := \nu x.\big((\mu y.x \wedge y) \vee \nu z.(z \wedge \mu y.x \wedge y)\big).$$

We leave it for the reader to verify that this formula has alternation depth two, and that its closure graph looks as follows:



Let α_y and α_z be the other two fixpoint formulas in the cluster of α_x, that is, let $\alpha_y := \mu y.\alpha_x \wedge y$ and $\alpha_z := \nu z.z \wedge \alpha_y$. These formulas correspond to the nodes in the graph that are labelled μy and νz, respectively. Now observe that we have $\alpha_x \triangleleft_f \alpha_y \triangleleft_f \alpha_z$, so that this cluster has an alternating $\sqsubset'_C$-chain of length *three*: $\alpha_z \sqsubset'_C \alpha_y \sqsubset'_C \alpha_x$. Note however, that any trace from α_y to α_z must pass through α_x, the $\sqsubset_C$-maximal element of the cluster. In particular, we do *not* have $\alpha_z \sqsubset_C \alpha_y$, so that there is *no* $\sqsubset_C$-chain of length three in the cluster. $\triangleleft$

Our first goal will be to prove the equivalence of any formula ξ to its associated parity formula $\mathbb{G}_\xi$, but for this purpose we need some auxiliary results. Our main technical lemma will be Proposition 6.34 below, which concerns the relation between the structures $\mathbb{G}_{\eta x.\chi}$ and $\mathbb{G}_\chi$. In order to prove this Proposition, we need some preliminary observations concerning the interaction of the notion of substitution with the operations of taking free subformulas and closure, respectively.

Proposition 6.32 *Let φ, ψ and ξ be formulas in μ*`ML` *such that $x \in FV(\varphi)$, and ξ is free for x in both φ and ψ. Then*

1) $\varphi \trianglelefteq_f \psi$ implies $\varphi[\xi/x] \trianglelefteq_f \psi[\xi/x]$;
2) $\varphi[\xi/x] \trianglelefteq_f \psi[\xi/x]$ implies $\varphi \trianglelefteq_f \psi$, provided that $\xi \ntrianglelefteq_f \varphi, \psi$.

Proposition 6.33 *Let ξ and χ be tidy μ-calculus formulas such that $BV(\chi) \cap FV(\xi) \neq \varnothing$ and $\chi[\xi/x]$ is tidy. Then the substitution operation $\xi/x : Clos(\chi) \to \mu\mathtt{ML}$ satisfies the following back- and forth condition, for every $\varphi \in Clos(\chi) \setminus \{x\}$:*

$$\{\chi \mid \varphi[\xi/x] \to_C \chi\} = \{\psi[\xi/x] \mid \varphi \to_C \psi\}. \tag{47}$$

Proposition 6.35 below states the equivalence of any tidy formula ξ to its associated parity formula $\mathbb{G}_\xi$. The proof of the main statement in this proposition proceeds by induction on the complexity of ξ, and the next proposition is the main technical ingredient in the key inductive step of this proof, where ξ is of the form $\eta x.\chi$. Roughly, Proposition 6.34 states that the substitution ξ/x is 'almost an isomorphism' between $\mathbb{G}_\chi$ and $\mathbb{G}_\xi$; note, however, that actually, rather than χ we consider its variant $\chi' := \chi[x'/x]$ — this guarantees tidyness. Recall that the alternation height $h^\downarrow(\xi)$ of a formula ξ was introduced in Definition 6.26.

Proposition 6.34 *Let $\xi = \eta x.\chi$ be a tidy fixpoint formula such that $x \in FV(\chi)$ and $\xi \notin Clos(\chi)$. Furthermore, let $\chi' := \chi[x'/x]$ for some fresh variable x'. Then χ' is tidy and the following hold.*
1) the substitution ξ/x' is a bijection between $Clos(\chi')$ and $Clos(\xi)$.
Let $\varphi, \psi \in Clos(\chi')$. Then we have
2) if $\varphi \neq x'$, then $\varphi \to_C \psi$ iff $\varphi[\xi/x'] \to_C \psi[\xi/x']$ and $L_C(\varphi) = L_C(\varphi[\xi/x'])$;
3) if $x' \in FV(\varphi)$ then $\varphi \trianglelefteq_f \psi$ iff $\varphi[\xi/x'] \trianglelefteq_f \psi[\xi/x']$;
4) if φ and ψ are fixpoint formulas then $\psi \sqsubseteq_C \varphi$ iff $\psi[\xi/x'] \sqsubseteq_C \varphi[\xi/x']$;
5) if $(\varphi_n)_{n\in\omega}$ is an infinite trace through $Clos(\chi')$, then $(\varphi_n)_{n\in\omega}$ has the same winner as $(\varphi_n[\xi/x'])_{n\in\omega}$.

Proof. Let $\xi = \eta x.\chi$ be a tidy fixpoint formula such that $x \in FV(\chi)$ and $\xi \notin Clos(\chi)$, and let $\chi' := \chi[x'/x]$ for some fresh variable x'. We leave it for the reader to verify that χ' is tidy, and first make the following technical observation:

$$\text{if } \varphi \in Clos(\chi') \text{ then } \xi \notin Clos(\varphi) \text{ and } \xi \ntrianglelefteq_f \varphi. \tag{48}$$

To see this, take an arbitrary $\varphi \in Clos(\chi')$, and first assume for contradiction that $\xi \in Clos(\varphi)$. Combining this with the assumption that $\varphi \in Clos(\chi')$ we get that $\xi \in Clos(\chi')$. By item 4) of Proposition 2.38 it holds that $Clos(\chi[x'/x]) = \{\rho[x'/x] \mid \rho \in Clos(\chi)\} \cup Clos(x')$. Thus, $\xi = \rho[x'/x]$ for some $\rho \in Clos(\chi)$ and because $x' \notin FV(\xi)$ it follows that $\xi = \rho \in Clos(\chi)$. But this contradicts the assumption that $\xi \notin Clos(\chi)$. In other words, we have proved that $\xi \notin Clos(\varphi)$. To see that also $\xi \ntrianglelefteq_f \varphi$ note that by Proposition 2.40 $\xi \trianglelefteq_f \varphi$ would entail $\xi \in Clos(\varphi)$.

We now turn to proving the respective items of the Proposition.

Item 1): We leave it for the reader to verify that the substitution ξ/x' is well-defined, i.e., that

$$\xi \text{ is free for } x' \text{ in every } \varphi \in Clos(\chi'), \tag{49}$$

and that $\varphi[\xi/x'] \in Clos(\xi)$, for all $\varphi \in Clos(\chi')$.

For injectivity of the substitution, suppose that $\varphi_0[\xi/x'] = \varphi_1[\xi/x']$, where $\varphi_0, \varphi_1 \in Clos(\chi')$. It follows by (48) that ξ is not a free subformula of either φ_0 or φ_1. But then it is immediate by Proposition 2.41 that $\varphi_0 = \varphi_1$.

For surjectivity, it suffices to show that ξ belongs to the set $\Phi := \{\varphi[\xi'/x] \mid \chi' \twoheadrightarrow_C \varphi\}$, and that the set Φ is closed, i.e., $\Phi \subseteq Clos(\Phi)$. But since we have $x' \in FV(\chi')$, we obtain $\chi' \twoheadrightarrow_C x'$ by Proposition 2.38(1), and so we have $\xi = x'[\xi/x'] \in \Phi$. The proof that Φ is closed is routine, and left as an exercise.

Item 2): This follows immediately from Proposition 6.33 and item 1). Note that the condition of Proposition 6.33 (viz., that $BV(\chi') \cap FV(\xi) = \varnothing$) follows because ξ is tidy and $BV(\chi') = BV(\chi) \subseteq BV(\xi)$, where the latter inclusion is item 1) of Proposition 2.35.

The claim that $L_C(\varphi) = L_C(\varphi[\xi/x'])$ is rather trivial.

Item 3) This is Proposition 6.32. The assumption $\xi \not\trianglelefteq_f \psi$ and $\xi \not\trianglelefteq_f \varphi$ follows from (48).

Item 4): For the left-to-right direction assume that $\psi \sqsubseteq_C \varphi$. By definition there is some trace $\varphi = \rho_0 \rightarrow_C \rho_1 \rightarrow_C \ldots \rightarrow_C \rho_n = \psi$ such that $\varphi \trianglelefteq_f \rho_i$ for all $i \in [0, \ldots, n]$. It is clear that none of the ρ_i is equal to x because x has no outgoing $\rightarrow_C$-edges and $\psi \neq x$. Thus we can use item 2) to obtain a trace $\varphi[\xi/x'] = \rho_0[\xi/x'] \rightarrow_C \rho_1[\xi/x'] \rightarrow_C \ldots \rightarrow_C \rho_n[\xi/x'] = \psi[\xi/x']$. By Proposition 6.32 it follows from $\varphi \trianglelefteq_f \rho_i$ that $\varphi[\xi/x'] \trianglelefteq_f \rho_i[\xi/x']$, for all $i \in [0, \ldots, n]$. That is, we have shown that $\varphi[\xi/x'] \twoheadrightarrow_C^{\varphi[\xi/x']} \psi[\xi/x']$.

Before we turn to the opposite direction we show that, for all $\rho, \sigma \in Clos(\chi')$, we have

$$\text{if } \rho[\xi/x'] \twoheadrightarrow_C \sigma[\xi/x'] \text{ and } x' \in FV(\sigma) \text{ then } x' \in FV(\rho). \tag{50}$$

This claim holds because, since ξ is free for x' in σ by (49), by definition of $\trianglelefteq_f$ it follows from $x' \in FV(\sigma)$ that $\xi \trianglelefteq_f \sigma[\xi/x']$, and thus we find $\sigma[\xi/x'] \twoheadrightarrow_C \xi$ by Proposition 2.40. If it were the case that $x' \notin FV(\rho)$ then we would have that $\rho = \rho[\xi/x'] \twoheadrightarrow_C \sigma[\xi/x'] \twoheadrightarrow_C \xi$, contradicting (48).

Turning to the right-to-left direction of item 4), assume that $\psi[\xi/x'] \sqsubseteq_C \varphi[\xi/x']$. This means that there is a trace $\varphi[\xi/x'] = \rho'_0 \rightarrow_C \ldots \rightarrow_C \rho'_m = \psi[\xi/x']$ with $\varphi[\xi/x'] \trianglelefteq_f \rho'_i$ for all $i \in [0, \ldots, m]$. By Proposition 6.25 we have $\psi[\xi/x'] \equiv_C \varphi[\xi/x']$. It follows from (50) and $\psi[\xi/x'] \equiv_C \varphi[\xi/x']$ that x' is either free in both φ and ψ, or free in neither of the two formulas. In the second case we obtain $\varphi = \varphi[\xi/x']$ and $\psi = \psi[\xi/x']$, so that the statement of this item holds trivially.

We now focus on the case where $x' \in FV(\varphi) \cap FV(\psi)$. Our first claim is that $\rho'_i \neq \xi$ for all $i \in [0, \ldots, m]$. This follows from the fact that $\varphi[\xi/x'] \trianglelefteq_f \rho'_i$, which holds by assumption, and the observation that ξ is a proper free subformula of $\varphi[\xi/x']$, which holds since φ is a fixpoint formula and hence, distinct from x'. But if $\rho'_i \neq \xi$ for all $i \in [0, \ldots, m]$, we may use the items 1) and 2) to obtain a trace $\varphi = \rho_0 \rightarrow_C \ldots \rightarrow_C \rho_m = \psi$ such that $\rho_i[\xi/x'] = \rho'_i$ for all $i \in [0, \ldots, m]$. Furthermore, by Proposition 2.35 it follows from $x' \in FV(\psi)$ that $x' \in FV(\rho_i)$, and so we may use item 3) to obtain $\varphi \trianglelefteq_f \rho_i$, for all $i \in [0, \ldots, m]$. This suffices to show that $\psi \sqsubseteq_C \varphi$.

Item 5): This observation is immediate by item 4) and Proposition 6.29. QED

Proposition 6.35 *Let ξ be a tidy μ-calculus formula. Then $\xi \equiv \mathbb{G}_\xi$.*

Proof. It will be convenient for us to consider the *global* formula graph $\mathbb{G} := (\mu\mathtt{ML}^t, \to_C, L_C, \Omega_g)$, where $\mu\mathtt{ML}^t$ is the set of all tidy formulas using a fixed infinite set of variables, and L_C is the obviously defined global labelling function. We may assign a semantics to this global graph using an equally obvious definition of an acceptance game, where the only non-standard aspect is that the carrier set of this 'formula' is infinite. For each tidy formula φ we may then consider the structure $\mathbb{G}\langle\varphi\rangle := (\mu\mathtt{ML}^t, \to_C, L_C, \Omega_g, \varphi)$ as an initialised (generalised) parity formula. Note that all structures of this form have the same (infinite) set of vertices, but that the only vertices that are accessible in $\mathbb{G}\langle\varphi\rangle$ are the formulas in the (finite) set $Clos(\varphi)$. It is then easy to see that $\mathbb{G}\langle\varphi\rangle \equiv \mathbb{G}_\xi\langle\varphi\rangle$, for any pair of tidy formulas φ, ξ such that $\varphi \in Clos(\xi)$.

In order to prove the Proposition, it therefore suffices to show that every tidy formula ξ satisfies the following:

$$\mathbb{G}\langle\varphi\rangle \equiv \varphi, \text{ for all } \varphi \in Clos(\xi). \tag{51}$$

We will prove (51) by induction on the length of ξ. In the base step of this induction we have $|\xi| = 1$, which means that ξ is an atomic formula. In this case it is easy to see that (51) holds.

In the induction step of the proof we assume that $|\xi| > 1$, and we make a case distinction. The cases where ξ is of the form $\xi = \xi_0 \odot \xi_1$ with $\odot \in \{\wedge, \vee\}$ or $\xi = \heartsuit\xi_0$ with $\heartsuit \in \{\Diamond, \Box\}$, are easy and left as exercises for the reader.

In the case where ξ is of the form $\xi = \eta x.\chi$ with $\eta \in \{\mu, \nu\}$ we make a further case distinction. If ξ belongs to the closure set of χ, then we have $Clos(\xi) \subseteq Clos(\chi)$, so that (51) immediately follows from the induction hypothesis, applied to the formula χ.

This leaves the case where ξ is of the form $\eta x.\chi$, while $\xi \notin Clos(\chi)$. Let x' be some fresh variable, then obviously we may apply the induction hypothesis to the (tidy) formula $\chi' := \chi[x'/x]$. The statement that $\xi \equiv \mathbb{G}\langle\xi\rangle$ now follows by a routine argument, based on the observations in Proposition 6.34. QED

It is left to show that the index of $\mathbb{G}_\xi$ does not exceed the alternation depth of the formula ξ. For this purpose it suffices to prove Proposition 6.39 below, which links the alternation hierarchy to the maximal length of alternating $\sqsubset_C$-chains. We need quite a bit of preparation to get there.

Our first auxiliary proposition states that, when analysing the alternation depth of a tidy formula of the form $\chi[\xi/x]$, we may without loss of generality assume that ξ is not a free subformula of χ. Recall that $ad_\eta(\xi)$ denotes the least k such that $\xi \in \Theta^\eta_k$.

Proposition 6.36 *Let ξ and χ be μ-calculus formulas such that ξ is free for x in χ, $x \in FV(\chi)$, $|\xi| > 1$, and $\chi[\xi/x]$ is tidy. Then there is a tidy formula χ' such that ξ is free for x' in χ', $\chi'[\xi/x'] = \chi[\xi/x]$, $|\chi'| \leq |\chi|$, $ad_\eta(\chi') \leq ad_\eta(\chi)$ for $\eta \in \{\mu, \nu\}$, and $\xi \not\trianglelefteq_f \chi'$.*

Our main auxiliary proposition concerns the relation between parity formulas of the form $\mathbb{G}_\chi$ and $\mathbb{G}_{\chi[\xi/x]}$, respectively. Roughly, it states that the substitution ξ/x is a 'local isomorphism' between these two structures, i.e., it is an isomorphism at the level of certain clusters. Recall that $C(\psi)$ denotes the $\equiv_C$-cluster of a formula ψ, cf. Definition 6.24.

Proposition 6.37 *Let ξ and χ be formulas such that ξ is free for x in χ, $\xi \not\trianglelefteq_f \chi$, and $x \notin FV(\xi)$. Furthermore, let $\psi \in Clos(\chi)$ be such that $\psi[\xi/x] \notin Clos(\chi) \cup Clos(\xi)$. Then the following hold:*

1) the substitution $\xi/x : C(\psi) \to C(\psi[\xi/x])$ is a bijection between $C(\psi)$ and $C(\psi[\xi/x])$.
Let $\varphi_0, \varphi_1 \in Clos(\chi')$. Then we have
2) $\varphi_0 \to_C \varphi_1$ iff $\varphi_0[\xi/x] \to_C \varphi_1[\xi/x]$ and $L_C(\varphi_0) = L_C(\varphi_0[\xi/x])$;
3) $\varphi_0 \trianglelefteq_f \varphi_1$ iff $\varphi_0[\xi/x] \trianglelefteq_f \varphi_1[\xi/x]$;
4) $h^\downarrow(\varphi_0) = h^\downarrow(\varphi_0[\xi/x])$, if φ_0 is a fixpoint formula.

The following Proposition is the key observation linking the alternation depth of a formula to the index of its associated automaton, and thus to the maximal length of alternating $\sqsubset_C$-chains in the closure graph of the formula. It is thus the result, announced at the end of section 2.6, that corresponds to Proposition 2.51 but applies to the wider class of tidy formulas.

To formulate and prove this observation, we need to refine some of our earlier definitions.

Definition 6.38 Let C be a closure cluster. For $\eta \in \{\mu, \nu\}$, define $cd_\eta(C)$ as the maximal length of an alternating $\sqsubset_C$-chain in C leading up to an η-formula. Given a formula ξ, let $cd_\eta(\xi)$ and $cd(\xi)$ be defined as the maximum value of $cd_\eta(C)$ and $cd(C)$, respectively, where C ranges over all clusters of $Clos(\xi)$. $\triangleleft$

Clearly then we have $cd(C) = \max(cd_\mu(C), cd_\nu(C))$, and, similarly, $cd(\xi) = \max(cd_\mu(\xi), cd_\nu(\xi))$.

Proposition 6.39 *For any tidy formula ξ and $\eta \in \{\mu, \nu\}$, we have*

$$cd_\eta(\xi) \leq n \text{ iff } \xi \in \Theta^\eta_n. \tag{52}$$

As a corollary, the alternation depth of ξ is equal to the length of its longest alternating $\sqsubset_C$-chain.

Proof. For the proof of the left-to-right direction of (52), we proceed by an outer induction on n, and an inner induction on the length $|\xi|$ of the formula ξ. We focus on the outer inductive case, leaving the base case, where $n = 0$, to the reader.

First of all, it is easy to see that every fixpoint formula ξ' in the cluster of ξ satisfies $cd_\eta(\xi') = cd_\eta(\xi)$, while it follows from Proposition 2.48 that $\xi' \in \Theta^\eta_n$ iff $\xi \in \Theta^\eta_n$. For this reason we may, without loss of generality, confine our attention to the case where ξ is the $\sqsubset_C$-maximal element of its cluster. Now distinguish cases, as to the parity of ξ.

First we consider the case where ξ is of the form $\xi = \overline{\eta}x.\chi$. Let

$$\eta_1 x_1.\psi_1 \sqsubset_C \eta_2 x_2.\psi_2 \sqsubset_C \cdots \sqsubset_C \eta_k x_k.\psi_k$$

be a maximal alternating η-chain in $Clos(\chi)$. Then

$$\eta_1 x_1.\psi_1[\xi/x] \sqsubset_C \eta_2 x_2.\psi_2 \sqsubset_C \cdots \sqsubset_C \eta_k x_k.\psi_k[\xi/x]$$

is an alternating η-chain in $Clos(\xi)$, and so we have $k \leq n$. It then follows by the inner induction hypothesis that $\chi \in \Theta^\eta_n$, and so by definition of the latter set we find $\xi = \overline{\eta}x.\chi \in \Theta^\eta_n$, as required.

The other case to be discussed is where ξ is of the form $\xi = \eta x.\chi$. Now let

$$\eta_1 x_1.\psi_1 \sqsubset_C \eta_2 x_2.\psi_2 \sqsubset_C \cdots \sqsubset_C \eta_k x_k.\psi_k$$

be a maximal alternating $\overline{\eta}$-chain in $Clos(\chi)$.

We now make a further case distinction. If x is a free variable of some formula in this chain, it is in fact a free variable of every formula in the chain; from this it follows that

$$\eta_1 x_1.\psi_1[\xi/x] \sqsubset_C \eta_2 x_2.\psi_2 \sqsubset_C \cdots \sqsubset_C \eta_k x_k.\psi_k[\xi/x] \sqsubset_C \xi$$

is an alternating η-chain in $Clos(\xi)$. Since this chain has length $k+1$, it follows by our assumption on ξ that $k+1 \leq n$, and so $k \leq n-1$. Alternatively, if x is not a free variable of any formula in this chain, then the chain is itself an alternating $\overline{\eta}$-chain in $Clos(\xi)$, and from this and the assumption that $cd_\eta(\xi) \leq n$ it readily follows that $k \leq n-1$.

In both cases we find that $k \leq n-1$, which means that $cd_{\overline{\eta}}(\chi) \leq n-1$. By the outer induction hypothesis we thus find that $\chi \in \Theta^{\overline{\eta}}_{n-1}$. From this it is then easy to derive that $\xi = \eta x.\chi \in \Theta^{\eta}_n$.

For a proof of the opposite, right-to-left direction '$\Leftarrow$' of (52), the argument proceeds by induction on the length of φ. In the base case φ is atomic and hence the claim is trivially true.

In the inductive step we make a case distinction depending on the clause of Definition 2.46 that was applied in the last step of the derivation of $\varphi \in \Theta^{\eta}_k$. We leave the easy cases, for the clauses 1 and 2, to the reader.

If clause 3 is used to derive $\varphi \in \Theta^{\eta}_n$ then $\varphi = \overline{\eta}x.\chi$ for some $\chi \in \Theta^{\eta}_n$. First define $\chi' = \chi[x'/x]$ for an x' that is fresh for χ and φ. Note that the length of χ' is equal to the length of χ, which is shorter than the length of φ, while obviously we also have that $\chi' \in \Theta^{\eta}_n$. Moreover, χ' is tidy because φ is tidy, $BV(\chi') = BV(\chi) \subseteq BV(\varphi)$, $FV(\chi') = (FV(\chi) \setminus \{x\}) \cup \{x'\} \subseteq FV(\varphi) \cup \{x'\}$, and x' is fresh for φ. This means that we can apply the inductive hypothesis to χ', obtaining that $cd_\eta(\chi') \leq n$

We then distinguish cases depending on whether $\varphi \in Clos(\chi)$ or not.

If $\varphi \in Clos(\chi)$ then it is not hard to prove that $\varphi \in Clos(\chi')$ as well. It is then easy to see that every alternating chain in $\mathbb{G}_\varphi$ also exists in $\mathbb{G}_{\chi'}$, and thus it follows that $cd_\eta(\varphi) \leq n$.

If $\varphi \notin Clos(\chi)$ we distinguish further cases depending on whether $x \in FV(\chi)$. If this is not the case then $\chi' = \chi$ and $\mathbb{G}_\varphi$ is just like $\mathbb{G}_\chi$ with an additional vertex for φ that forms a degenerate cluster on its own and is connected just with an outgoing $\to_C$-edge to the vertex of χ' in $\mathbb{G}_{\chi'}$. Thus, every alternating chain in a cluster of $\mathbb{G}_\varphi$ also exists in $\mathbb{G}_{\chi'}$ and thus $cd_\eta(\varphi) \leq n$ follows from $cd_\eta(\chi') \leq n$.

The last case is where $\varphi \notin Clos(\chi)$ and $x \in FV(\chi)$. To prove $cd_\eta(\varphi) \leq n$ consider an alternating $\sqsubset_C$-chain $\eta_1 x_1.\rho_1 \sqsubset_C \cdots \sqsubset_C \eta_m x_m.\rho_m$, of length m and with $\eta_m = \eta$ in some cluster of $\mathbb{G}_\varphi$. We now argue that $m \leq n$. Because $\eta_i x_i.\rho_i \in Clos(\varphi)$ for all $i \in [1, \ldots, m]$ it follows by Proposition 6.30 that the only possibility for φ to be among the $\eta_i x_i.\rho_i$ in this chain is if $\varphi = \eta_m x_m.\rho_m$. This would lead to a contradiction however, because $\eta_m = \eta$ while we assumed that $\varphi = \overline{\eta}x.\chi$. We may therefore conclude that φ is not among the $\eta_i x_i.\rho_i$ for $i \in [1, \ldots, m]$. By the items 1), 2) and 4) of Proposition 6.34 it follows that there is an alternating

$\sqsubset_C$-chain $\eta_1 x_1.\sigma_1 \sqsubset_C \cdots \sqsubset_C \eta_m x_m.\sigma_m$ in $Clos(\chi')$ such that $(\eta_i x_i.\sigma_i)[\xi/x'] = \eta_i x_i.\rho_i$ for all $i \in [1,\dots,m]$. Because $cd_\eta(\chi') \leq n$ it follows that $m \leq n$.

If clause 4 is used to derive $\varphi \in \Theta^\eta_n$ then φ is of the form $\varphi = \chi[\xi/x]$ such that $\chi, \xi \in \Theta^\eta_n$. First observe that we may assume that $x \in FV(\chi)$ and $|\xi| > 1$, otherwise the claim trivialises. Furthermore, because of Proposition 6.36 we may without loss of generality assume that in addition χ is tidy as well, that x is fresh for ξ, and that $\xi \ntrianglelefteq_f \chi$. Finally, since $|\xi| > 1$ we find that the length of χ is smaller than that of $\varphi = \chi[\xi/x]$, so that we may apply the inductive hypothesis, which gives that $cd_\eta(\chi) \leq n$ and $cd_\eta(\xi) \leq n$.

To show that $cd_\eta(\chi[\xi/x]) \leq n$ clearly it suffices to prove that $h^\downarrow(\eta y.\rho) \leq n$, for any fixpoint formula $\eta x.\rho \in Clos(\chi[\xi/x])$ that is at the top of a maximal alternating $\sqsubset_C$-chain in $\mathbb{G}_{\chi[\xi/x]}$. The key claim here is that

$$h^\downarrow(\lambda y.\rho) = h^\downarrow(\lambda y.\rho') \text{ for some } \lambda y.\rho' \in Clos(\chi) \cup Clos(\xi). \tag{53}$$

To see this, first note that we may assume that $\lambda y.\rho \notin Clos(\chi) \cup Clos(\xi)$ because otherwise we can just set $\rho' := \rho$. By Proposition 2.38 we obtain that

$$Clos(\chi[\xi/x]) = \{\psi[\xi/x] \mid \psi \in Clos(\chi)\} \cup Clos(\xi).$$

Therefore, since $\lambda y.\rho \in Clos(\chi[\xi/x])$, and we assume that $\lambda y.\rho \notin Clos(\xi)$, it follows that $\lambda y.\rho = \psi[\xi/x]$ for some $\psi \in Clos(\chi)$. We are thus in a position to apply Proposition 6.37, which describes how the $\rightarrow_C$-cluster of ψ relates under the substitution ξ/x to the $\rightarrow_C$-cluster of $\lambda y.\rho = \psi[\xi/x]$. Note that $\psi \neq x$ because otherwise we would have $\lambda y.\rho = \xi$, contradicting the assumption that $\lambda x.\rho \notin Clos(\xi)$. This means that $\psi = \lambda y.\rho'$ for some formula ρ', since by item 2) of Proposition 6.37 the substitution ξ/x preserves the main connective of formulas other than x. Finally, it follows from item 4) of Proposition 6.37 that $h^\downarrow(\lambda y.\rho) = h^\downarrow(\lambda y.\rho')$.

As an immediate consequence of (53) we obtain that $h^\downarrow(\eta y.\rho') \leq n$ because $\eta y.\rho'$ is either in $\mathbb{G}_\chi$ or in $\mathbb{G}_\xi$, where the inductive hypothesis applies. This finishes the proof for the case of clause 4.

We leave the last case, where clause 5 is used to derive that $\varphi \in \Theta^\eta_n$, to the reader. QED

Now that we have proved the main technical lemma, our desired result about the index of the parity formula $\mathbb{G}_\xi$ is almost immediate.

Proposition 6.40 *For every tidy formula ξ it holds that $ind(\mathbb{G}_\xi) \leq ad(\xi)$.*

Proof. Take an arbitrary fixpoint formula ξ, and assume that $ad(\xi) \leq n$. Clearly it suffices to show that $ind(\mathbb{G}_\xi) \leq n$.

For this purpose, first observe that by $ad(\xi) \leq n$ we find that $\xi \in \Theta^\mu_n \cap \Theta^\nu_n$. Then it follows by Proposition 6.39 that $cd(\xi) \leq n$, so that by Proposition 6.28 we obtain $ind(\mathbb{G}_\xi) \leq n$. QED

6.4 Guarded transformation

As an example of an important construction on parity formulas, we consider the operation of guarded transformation. Recall from Definition 2.15 that a μ-calculus formula is *guarded* if every occurrence of a bound variable is in the scope of a modal operator which resides inside the variable's defining fixpoint formula. Intuitively, the effect of this condition is that, when evaluating a guarded formula in some model, between any two iterations of the same fixpoint variable, one has make a transition in the model. Many constructions and algorithms operating on μ-calculus formulas presuppose that the input formula is in guarded form, which explains the need for low-cost *guarded transformations*, that is, efficient procedures for bringing a μ-calculus formula into an equivalent guarded form.

It is easy to translate the notion of guardedness to parity formulas, but in fact we will need something stronger in the next chapter, when we present the automata-theoretic perspective on the modal μ-calculus.

Definition 6.41 A path $\pi = v_0 v_1 \cdots v_n$ through a parity formula is *unguarded* if $n \geq 1$, $v_0, v_n \in \mathsf{Dom}(\Omega)$ while there is no i, with $0 < i \leq n$, such that v_i is a modal node. A parity formula is *guarded* if it has no unguarded cycles, and *strongly guarded* if it has no unguarded paths. ◁

In words, a parity formula is strongly guarded if every path, leading from one state (node in $\mathsf{Dom}(\Omega)$) to another contains at least one modal node (occurring after the path's starting state). The following theorem states that on arbitrary parity formulas, we can give an exponential-size guarded transformation; note that the index of the formula does not change. At the time of writing it is not known whether every parity formula can be transformed into a guarded equivalent of *subexponential size.*

Theorem 6.42 *There is an algorithm that transforms a parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$ into a strongly guarded parity formula $\mathbb{G}^g$ such that*

1) $\mathbb{G}^g \equiv \mathbb{G}$*;*
2) $|\mathbb{G}^g| \leq 2^{1+|\mathsf{Dom}(\Omega)|} \cdot |\mathbb{G}|$*;*
3) $ind(\mathbb{G}^g) \leq ind(\mathbb{G})$*;*

We will prove Theorem 6.42 via a construction that step by step improves the 'degree of guardedness' of the parity formula. In the intermediate steps we will be dealing with a modified notion of guardedness.

Definition 6.43 A parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$ is *strongly k-guarded* if it every unguarded path $\pi = v_0 v_1 \cdots v_n$ satisfies $\Omega(v_n) > k$. ◁

Clearly, a parity formula is (strongly) guarded iff it is (strongly) m-guarded, where m is the maximum priority value of the formula. Hence, we may prove Theorem 6.42 by successively applying the following proposition. Recall that a parity formula is called *lean* if its priority map is injective. We say that a parity formula has *silent states only* if each of its states is labelled ϵ.

Proposition 6.44 *Let $\mathbb{G}$ be a lean, strongly k-guarded parity formula with silent states only. Then we can effectively obtain a lean, $k+1$-guarded parity formula $\mathbb{G}'$, with silent states only, and such that $\mathbb{G}' \equiv \mathbb{G}$, $|\mathbb{G}'| \leq 2 \cdot |\mathbb{G}|$ and $ind(\mathbb{G}') \leq ind(\mathbb{G})$.*

Proof. Let $\mathbb{G} = (V, E, L, \Omega, v_I)$ be an arbitrary lean, strongly k-guarded parity formula with silent states, that is, $\mathsf{Dom}(\Omega) \subseteq L^{-1}(\epsilon)$. Without loss of generality we may assume that in fact $\mathsf{Dom}(\Omega) = L^{-1}(\epsilon)$. If $\mathbb{G}$ happens to be already $k+1$-guarded, then there is nothing to do: we may simply define $\mathbb{G}' := \mathbb{G}$.

On the other hand, if $\mathbb{G}$ is $k+1$-unguarded, then in particular there must be a state $z \in V$ such that $\Omega(z) = k+1$. By injectivity of Ω, z is unique with this property. In this case we will build the parity formula $\mathbb{G}'$, roughly, on the disjoint union of $\mathbb{G}$, a copy of a part of $\mathbb{G}$ that is in some sense generated from z, and an additional copy of z itself.

For the definition of $\mathbb{G}'$, let W^z be the smallest set $W \subseteq V$ containing z, which is such that $E[w] \subseteq W$ whenever $w \in W$ is boolean. Now define

$$V' := \big(V \times \{0\}\big) \cup \big(W^z \times \{1\}\big) \cup \big(\{z\} \times \{2\}\big).$$

In the sequel we may write u_0 instead of $(u, 0)$, for brevity. Furthermore, recall that we use V_m to denote the set of *modal* vertices of $\mathbb{G}$. The edge relation E' is now given as follows:

$$\begin{array}{rlcl}
E' := & \big\{(u_0, v_0) \mid (u,v) \in E \text{ and } v \neq z\big\} & \cup & \big\{(u_1, v_1) \mid (u,v) \in E \text{ and } v \neq z\big\} \\
\cup & \big\{(u_0, z_1) \mid (u,z) \in E\big\} & & \\
\cup & \big\{(u_1, v_0) \mid (u,v) \in E \text{ and } u \in V_m\big\} & \cup & \big\{(u_1, u_0) \mid u \in \mathsf{Dom}(\Omega) \text{ and } \Omega(u) > k+1\big\} \\
\cup & \big\{(u_1, z_2) \mid (u,z) \in E \text{ and } u \notin V_m\big\} & &
\end{array}$$

To understand the graph (V', E'), it helps, first of all, to realise that the set W^z provides a subgraph of (V, E), which forms a dag with root z and such that every 'leaf' is either a modal or propositional node, or else a state $v \in \mathsf{Dom}(\Omega)$ with $\Omega(v) > k$. (It cannot be the case that $\Omega(v) \leq k$ due to the assumed k-guardedness of $\mathbb{G}$.) Second, it is important to realise that the only way to move from the V-part of V' to the W^z-part is via the root z_1 of the W^z-part, while the only way to move in the converse direction is either directly following a modal node, or else by making a dummy transition from some vertex u_1 to its counterpart u_0 for any $u \in W^z$ with $\Omega(u) > k$. Finally, we add a single vertex z_2 to V'.

Furthermore, we define the labelling L' and the priority map Ω' of $\mathbb{G}'$ by putting

$$L'(u_i) := \begin{cases} L(u) & \text{if } i = 0,1 \\ \widehat{z} & \text{if } u_i = z_2 \end{cases}$$

where we recall that $\widehat{z} = \bot$ if $\Omega(z)$ is odd and $\widehat{z} = \top$ if $\Omega(z)$ is even, and

$$\Omega'(u_i) := \begin{cases} \Omega(u) & \text{if } i = 0 \text{ and } u \in \mathsf{Dom}(\Omega) \\ \uparrow & \text{otherwise.} \end{cases}$$

In words, the label of a node (v, i) in $\mathbb{G}'$ is identical to the one of v in $\mathbb{G}$, with the sole exception of the vertex $(z, 2)$. To explain the label of the latter node, note that by construction, any unguarded E'-path from z_1 to z_2 projects to an unguarded $k+1$-cycle from

z to z in $\mathbb{G}$. If $\Omega(z) = k + 1$ is odd, any such cycle represents (tails of) infinite matches that are lost by $\exists$; for this reason we may label the 'second' appearance of z in the E'-path, i.e., as the node z_2, with $\perp$.

We now turn to the proof of the proposition. It is not hard to show that $\mathbb{G}'$ is lean and that $|\mathbb{G}'| \leq 2 \cdot |\mathbb{G}|$.

To show that $ind(\mathbb{G}') \leq ind(\mathbb{G})$, note that obviously, the projection map $u_i \mapsto u$ preserves the cluster equivalence relation, i.e., $u_i \equiv_{E'} v_j$ implies $u \equiv_E v$. Hence, the image of any cluster C' of $\mathbb{G}'$ under this projection is part of some cluster C of $\mathbb{G}$. But then by definition of Ω' it is easy to see that $ind(C') \leq ind(C)$. From this it is immediate that $ind(\mathbb{G}') \leq ind(\mathbb{G})$.

To see why $\mathbb{G}'$ is $k+1$-guarded, suppose for contradiction that it has a $k+1$-unguarded path $\pi = (v_0, i_0)(v_1, i_1) \cdots (v_n, i_n)$. It is easy to see that this implies that the *projection* $v_0 v_1 \cdots v_n$ of π is an unguarded path in $\mathbb{G}$ (here we ignore possible dummy transitions of the form (u_1, u_0)), and so by assumption on $\mathbb{G}$ it must be the case that $\Omega'(v_n, i_n) = \Omega(v_n) = k+1$. This means that $(v_n, i_n) = (z, 0)$; but the only way to arrive at the node $(z, 0)$ in (V', E') is directly following a modal node (in $W^z \times \{1\}$), which contradicts the unguardedness of the path π.

In order to finish the proof of the Proposition, we need to prove the equivalence of $\mathbb{G}'$ and $\mathbb{G}$; but this can be established by a relatively routine argument of which we skip the details. QED

Proof of Theorem 6.42. Let $\mathbb{G}$ be an arbitrary parity formula; without loss of generality we may assume that $\mathbb{G}$ is lean, i.e., Ω is injective. Let $\mathsf{Ran}(\Omega) = \{k_1, \ldots, k_n\}$; then $|\mathsf{Dom}(\Omega)| = n$. To ensure that all states are silent, we may have to duplicate some vertices; that is, we continue with a version $\mathbb{H}$ of $\mathbb{G}$ that has at most twice as many vertices, but the same index, the same number of states, and silent state only.

By a straightforward induction we apply Proposition 6.44 to construct, for every $i \in [1, \ldots, n]$, a k_i-guarded parity automaton $\mathbb{H}^i$ with silent states only, and such that $\mathbb{H}^i \equiv \mathbb{G}$, $|\mathbb{H}^i| \leq 2^{i+1} \cdot |\mathbb{G}|$, and $ind(\mathbb{H}^i) = ind(\mathbb{G})$. Clearly then we find that $\mathbb{H}^n$ is the desired strongly guarded equivalent of $\mathbb{G}$; and since $n = |\mathsf{Dom}(\Omega)|$ we find that $|\mathbb{H}^n| \leq 2^{1+n} \cdot |\mathbb{G}|$ as required. QED

Remark 6.45 On a closer inspection of the construction in the proof of Proposition 6.44, the reader may observe that inductively, we may assume that for every i, every predecessor of a state in $\mathbb{H}^i$ with priority at most k_i is in fact a modal node. From this, it follows that we may impose, in the formulation of Theorem 6.42, an additional constraint on $\mathbb{G}^g$, namely, that every predecessor of a state is a modal node, more formally, that $(E^g)^{-1}[\mathsf{Dom}(\Omega] \subseteq V^g_m$. $\lhd$

6.5 From parity formulas to regular formulas

In section 6.3 we saw constructions that, for a given regular formula, produce equivalent parity formulas based on, respectively, the subformula dag and the closure graph of the original formula. We will now move in the opposite direction: we will give a construction that turns an arbitrary parity formula $\mathbb{G}$ into an equivalent regular formula $\xi_{\mathbb{G}} \in \mu\mathtt{ML}$. Basically this construction takes a parity formulas as a system of equations, and it solves these equations by a Gaussian elimination of variables. As a result, the transformation from parity formulas to regular formulas can be seen as some sort of *unravelling* construction.

Interestingly, we encounter a significant difference between the two size measures introduced in section 2.5: whereas the closure-size of the resulting formula $\xi_{\mathbb{G}}$ is *linear* in the size of $\mathbb{G}$, its number of subformulas is only guaranteed to be exponential. And in fact, Proposition 6.50 shows that there is a family of parity formulas for which the translation actually reaches this exponential subformula-size.

Proposition 6.46 *There is an effective procedure providing for any parity formula* $\mathbb{G} = (V, E, L, \Omega, v_I)$ *over some set* P *of proposition letters, a map* $\mathtt{tr}_{\mathbb{G}} : V \to \mu\mathtt{ML}(\mathsf{P})$ *such that*

1) $\mathbb{G}\langle v\rangle \equiv \mathtt{tr}_{\mathbb{G}}(v)$*, for every* $v \in V$*;*
2) $|\mathtt{tr}_{\mathbb{G}}(v)| \leq 2 \cdot |\mathbb{G}|$*;*
3) $|\mathit{Sfor}(\mathtt{tr}_{\mathbb{G}}(v))|$ *is at most exponential in* $|\mathbb{G}|$*;*
4) $\mathit{ad}(\mathtt{tr}_{\mathbb{G}}(v_I)) \leq \mathit{ind}(\mathbb{G})$*.*

Clearly, the algorithm mentioned in the Theorem will produce, given a parity formula $\mathbb{G} = (V, E, L, \Omega, v_I)$, an equivalent μ-calculus formula. Note that, although the definition of the translation map $\mathtt{tr}_{\mathbb{G}}$ involves many substitution operations, it does *not* involve any renaming of variables.

Definition 6.47 Let $\mathbb{G} = (V, E, L, \Omega, v_I)$ be a parity formula over some set P of proposition letters. We define

$$\xi_{\mathbb{G}} := \mathtt{tr}_{\mathbb{G}}(v_I)$$

and call $\xi_{\mathbb{G}}$ the μ-calculus formula *associated with* $\mathbb{G}$. $\lhd$

The following theorem is an immediate corollary of Proposition 6.46.

Corollary 6.48 *Let* $\mathbb{G}$ *be a parity formula. Then we find* $\xi_{\mathbb{G}} \equiv \mathbb{G}$*,* $|\xi_{\mathbb{G}}| \leq |\mathbb{G}|$ *and* $\mathit{ad}(\xi_{\mathbb{G}}) \leq \mathit{ind}(\mathbb{G})$*.*

Remark 6.49 Note that in item 3) of Proposition 6.46 we cannot state that the *dag-size* of $\mathtt{tr}_{\mathbb{G}}$ is at most exponential in the size of $\mathbb{G}$ since the formula $\mathtt{tr}_{\mathbb{G}}$ will generally not be clean, and so its dag-size may not be defined. For this reason we compare the number of subformulas of $\mathtt{tr}_{\mathbb{G}}$ to the size of $\mathbb{G}$. $\lhd$

▶ `construction as mentioned in Proposition 6.46 to be supplied.`

Proposition 6.50 *There is a family* $(\mathbb{F}_n)_{n\in\omega}$ *such that for every* n *it holds that* $|\mathbb{F}_n| \leq 2n+2$*, which implies that* $|\xi_{\mathbb{F}_n}|$ *is linear in* n*, while* $|\mathit{Sfor}(\xi_n| \geq 2^n$*.*

Notes

The structures that we call 'parity formulas' are in fact tightly related to the *alternating tree automata* introduced by Wilke [28], and also to so-called hierarchical equation systems (see for instance [2, 9], and references therein). Theorem 6.20 is essentially a reformulation of Wilke [28, Theorem 1]. Proposition 6.50 is essentially due to Bruse, Friedmann & Lange [5].

▶ `More notes to be supplied.`

Exercises

Exercise 6.1 Prove Proposition 6.30.

Exercise 6.2 (guarded transformation) Prove the equivalence of the parity formulas $\mathbb{G}$ and $\mathbb{G}'$ in the proof of Proposition 6.44.

7 Modal automata

7.1 Introduction

In this chapter we introduce and discuss the automata that we shall use to study the modal μ-calculus. These automata come in various shapes and types, but they all operate on the same type of structures, namely pointed Kripke structures, or transition systems.

The basic idea is that automata can be seen as alternatives to formulas. In particular, an automaton $\mathbb{A}$ will either accept of reject a given pointed Kripke model, and thus it can be compared to a formula ξ, which will either be true or false at a point in a Kripke model. This inspires the following definition.

Definition 7.1 Let $\mathbb{A}$ be an automaton, and assume that we have defined the notions of acceptance and rejection of a pointed Kripke model by such an automaton. In case $\mathbb{A}$ *accepts* the pointed Kripke structure $(\mathbb{S}, s)$ we write $\mathbb{S}, s \Vdash \mathbb{A}$, and *rejection* of $(\mathbb{S}, s)$ by $\mathbb{A}$ is denoted as $\mathbb{S}, s \nVdash \mathbb{A}$. The class of pointed Kripke models that are accepted by a given automaton $\mathbb{A}$ is denoted as $\mathcal{Q}(\mathbb{A})$, and we will sometimes refer to $\mathcal{Q}(\mathbb{A})$ as the class or *query* that is *recognized* by $\mathbb{A}$. Two automata $\mathbb{A}$ and $\mathbb{A}'$ are *equivalent*, notation: $\mathbb{A} \equiv \mathbb{A}'$, if $\mathcal{Q}(\mathbb{A}) = \mathcal{Q}(\mathbb{A}')$.

We say that a formula ξ is *equivalent* to $\mathbb{A}$, notation: $\xi \equiv \mathbb{A}$, if $\mathbb{S}, s \Vdash \xi$ iff $\mathbb{A}$ accepts $(\mathbb{S}, s)$, for every pointed Kripke model $(\mathbb{S}, s)$. $\lhd$

All our automata will be of the form $\mathbb{A} = \langle A, \Theta, Acc, a_I \rangle$ where A is a finite set of states, $Acc \subseteq A^\omega$ is the acceptance condition (usually given by a parity map Ω), $a_I \in A$ is the starting state of the automaton, and the transition map Θ has as its domain the set $A \times C$, where $C = \wp(\mathsf{P})$ is the set of *colors* over some set P of proposition letters. We will almost exclusively work with automata that are themselves *logic-based*, in the sense that the co-domain of Θ is some logical language consisting of relatively simple *one-step* formulas over the carrier set A of the automata. In other words, the states in A will play a double role as propositional *variables*.

For each type of automaton that we will encounter, the question whether such a device accepts or rejects a given pointed Kripke model $(\mathbb{S}, s)$ is determined by playing some kind of infinite board game that we call the *acceptance game* associated with the automaton and the Kripke structure. This game will always proceed in *rounds*, each of which starts and ends at a so-called *basic position* $(a, s) \in A \times S$, and consists of the two players, $\exists$ and $\forall$, moving a token via some intermediate position(s) to a new basic position. For a rough, intuitive understanding of the acceptance game, the reader may think of $\exists$ claiming, at a basic position (a, s), that the automaton $\mathbb{A}$, taken from the perspective a, is a good 'description' of the pointed structure $(\mathbb{S}, s)$.

The rules of the game are determined by the precise shape of the transition function Θ, and in each case will be given explicitly. The winning conditions of the acceptance game are fixed. Finite matches, as always, are lost by the player who got stuck. The winner of an infinite match Σ is always determined by applying the acceptance condition Acc to the infinite A-stream $a_I a_1 a_2 \cdots$ which is induced by the sequence $(a_I, s)(a_1, s_1)(a_2, s_2) \cdots$ of basic positions occurring in Σ. 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.

To understand the connections between the various kinds of automata, it is good to understand *how* one round of the game takes a match from one basic position (a_i, s_i) to the next (a_{i+1}, s_{i+1}). In principle, it is $\exists$'s task to propose a set $Z_i \subseteq A \times S$ of *witnesses* that substantiate her claim that the automaton $\mathbb{A}$, taken from the perspective a_i, is a good description of the pointed model $(\mathbb{S}, s_i)$. Then it is $\forall$ who picks the new basic position (a_{i+1}, s_{i+1}) as an element of the set Z_i. In fact, all acceptance games featuring in this chapter could be formulated in such a way that these are exactly the moves that players can make. However, we will usually take a slightly different perspective on the witness relation. In particular, since we are often thinking of A as a set of propositional variables, it will make sense to represent a relation $Z \subseteq A \times S$ as either a *valuation* $V_Z : A \to \wp S$ or as a *marking* or *coloring* $m_Z : S \to \wp A$, defined by putting, respectively,

$$\begin{array}{lcl} V_Z(a) & := & \{s \in S \mid (a, s) \in Z\} \\ m_Z(s) & := & \{a \in A \mid (a, s) \in Z\}. \end{array}$$

As already mentioned, the automata that we shall meet here come in various shapes, and they can be classified in many ways. One crucial distinction to make is that between *alternating* and *non-deterministic* automata. Where the generic modal automaton that we will introduce here is of the alternating type, many results on the modal μ-calculus are proved using the subclass of non-deterministic automata, where the transition map is of a conceptually simpler kind. What makes an automaton *nondeterministic* is the interaction pattern between the two players in the acceptance game: when the automaton is non-deterministic, a winning strategy for $\exists$ should in principle (but depending on the branching structure of the transition system) reduce the role of $\forall$ to that of a *path finder* in the model.

▶ `For the time being we restrict attention to the mono-modal case.`

7.2 Modal automata

Modal automata are based on the *modal one-step language*. This language consists of very simple modal formulas, built up from a collection A of propositional *variables*, corresponding to the bound variables of a formula.

Definition 7.2 Given a set X, we define the set $\mathtt{Latt}(X)$ of *lattice terms* over X through the following grammar:

$$\pi \ ::= \ \bot \mid \top \mid x \mid \pi \wedge \pi \mid \pi \vee \pi,$$

where $x \in X$. Given a set A, we define the set $\mathtt{1ML}(A)$ of *modal one-step formulas* over A by the following grammar:

$$\alpha \ ::= \ \bot \mid \top \mid \Diamond\pi \mid \Box\pi \mid \alpha \wedge \alpha \mid \alpha \vee \alpha,$$

with $\pi \in \mathtt{Latt}(A)$. ◁

Examples of one-step formula are $\Diamond(a \wedge b)$ or $\Box\bot \vee (\Diamond a \wedge \Box b)$. Observe that the set of modal one-step formulas over A corresponds to the set of lattice terms over the set $\{\Diamond\pi, \Box\pi \mid \pi \in \mathtt{Latt}(A)\}$. Observe too that every occurrence of an element of A must be positive, and in the scope of exactly one modality.

Definition 7.3 A *modal* P-*automaton* $\mathbb{A}$ is a quadruple (A, Θ, Ω, a_I) where A is a non-empty finite set of *states*, of which $a_I \in A$ is the *initial* state, $\Omega : A \to \omega$ is the *priority map*, and the *transition map*

$$\Theta : A \times \wp\mathsf{P} \to \mathtt{1ML}(A)$$

maps states to one-step formulas. The class of modal automata over the set P is denoted as $Aut_{\mathsf{P}}(\mathtt{1ML})$. $\lhd$

The operational semantics of modal automata is defined in terms of a so-called *acceptance game* $\mathcal{A}(\mathbb{A}, \mathbb{S})$ associated with a modal automaton $\mathbb{A}$ and a Kripke structure $\mathbb{S}$. $\exists$'s moves in this game will consist of 'local' valuations for the propositional variables in A, or rather, *markings* $m : S \to \wp A$. Such a marking turns a Kripke model over P into a Kripke model over the set $\mathsf{P} \cup A$.

Throughout this chapter we will represent a Kripke model (S, R, V) *coalgebraically* as a triple (S, R, σ_V) where we think of the binary relation R as a map $R : S \to \wp(S)$, and represent the valuation $V : \mathsf{P} \to \wp(S)$ as its transpose colouring $\sigma_V : S \to \wp(\mathsf{P})$.

Definition 7.4 Let P and A be disjoint sets of proposition letters and propositional variables, respectively. Given a Kripke model $\mathbb{S} = (S, R, \sigma_V)$ over the set P, and an A-marking $m : S \to \wp A$, we let $\mathbb{S} \oplus m$ denote the Kripke model $(S, R, \sigma_V \cup m)$, where $\sigma_V \oplus m$ is the marking given by $\sigma_V \oplus m(s) := \sigma_V(s) \cup m(s)$. $\lhd$

Definition 7.5 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 the parity game that is determined by the rules given in Table 9. Positions of the form $(a, s) \in A \times S$ are called *basic*. $\lhd$

Position	Player	Admissible moves	Priority
$(a, s) \in A \times S$	$\exists$	$\{m : S \to \wp A \mid \mathbb{S} \oplus m, s \Vdash \Theta(a, \sigma_V(s))\}$	$\Omega(a)$
$m : S \to \wp A$	$\forall$	$\{(b, t) \mid b \in m(t)\}$	0

Table 9: Acceptance game for modal automata

As explained in the introduction to this chapter, matches of the acceptance game proceed in *rounds*, moving from one basic position to the next. During a round of the game, the players are inspecting a local 'window' into the Kripke model, by means of a one-step formula. Concretely, at the start of a round, $\exists$'s task at a basic position (a, s) is to *satisfy* the one-step formula $\Theta(a, \sigma_V(s))$ at the state s in $\mathbb{S}$. For this purpose, she has to come up with a interpretation for the variables in A, since this is not provided by the valuation V of $\mathbb{S}$. More specifically, $\exists$ has to select a marking $m : S \to \wp A$, in such a way that the formula $\Theta(a, \sigma_V(s))$

becomes true at s in the model $\mathbb{S}\oplus m$ (as given in Definition 7.4). Once $\exists$ has made her choice, it is $\forall$'s turn; he needs to pick a new basic position from the witness set $\{(b,t) \mid b \in m(t)\}$.

Observe that both players could get stuck in such a match. For instance, it might be impossible for $\exists$ to satisfy the formula $\Theta(a, \sigma_V(s))$ at the state s, because the formula requires s to have successors where it has none. Alternatively, if $\exists$ could pick the *empty* marking m at a position (a,s), then she would immediately win the match since $\forall$ would get stuck.

▶ `examples of modal automata`

Remark 7.6 Note that it is in $\exists$'s interest to keep, at any basic position (s,a) of the acceptance game, the set of witnesses as small as possible. More precisely, if at some position (a,s) of the game, $\exists$ has two admissible markings, say, m and m', at her disposal, and these are such that $Z_m := \{(b,t) \in S \times A \mid b \in m(t)\} \subseteq Z_{m'} := \{(b,t) \in S \times A \mid b \in m'(t)\}$, then it will always be to her advantage to choose the marking m rather than m'. In particular, since all occurrences of propositional variables from A in one-step formulas must be in the scope of exactly one modality, to satisfy such a formula at a given point s of the model, the only points that matter are the successors of s. For these reasons, we may without loss of generality restrict the admissible moves of $\exists$ at a position (a,s) of the acceptance game to those markings m of which the domain is the collection of successors of current point s. In section 7.4 we will work out this perspective. $\lhd$

Convention 7.7 We will usually identify a match $\Sigma = (a_0,s_0)m_0(a_1,s_1)m_1(a_2,s_2)m_2\ldots$ of the acceptance game $\mathcal{A}(\mathbb{A},\mathbb{S})$ with the sequence $(a_0,s_0)(a_1,s_1)(a_2,s_2)\ldots$ of its *basic positions.*

Some basic concepts concerning modal automata are introduced in the following definition.

Definition 7.8 Fix a modal P-automaton $\mathbb{A} = (A, \Theta, \Omega, a_I)$.

Given a state a of $\mathbb{A}$, we write $\eta_a = \mu$ if $\Omega(a)$ is odd, and $\eta_a = \nu$ if $\Omega(a)$ is even; we call η_a the *(fixpoint) type* of a and say that a is an η_a-state. The sets of μ- and ν-states are denoted with A^μ and A^ν, respectively.

The *occurrence graph* of $\mathbb{A}$ is the directed graph $(G, E_{\mathbb{A}})$, where $E_{\mathbb{A}}ab$ if b occurs in $\Theta(a,c)$ for some $c \in \wp(\mathsf{P})$. We let $\lhd_{\mathbb{A}}$ denote the transitive closure of the converse relation $E_{\mathbb{A}}^{-1}$ of $E_{\mathbb{A}}$ and say that b is *active* in a if $b \lhd_{\mathbb{A}} a$. We write $a \bowtie_{\mathbb{A}} b$ if $a \lhd_{\mathbb{A}} b$ and $b \lhd_{\mathbb{A}} a$. A *cluster* of $\mathbb{A}$ is a cell of the equivalence relation generated by $\bowtie_{\mathbb{A}}$ (i.e., the smallest equivalence relation on A containing $\bowtie_{\mathbb{A}}$); a cluster C is *degenerate* if it is of the form $C = \{a\}$ with $a \not\bowtie_{\mathbb{A}} a$. The unique cluster to which a state $a \in A$ belongs is denoted as C_a. We write $a \sqsubset_{\mathbb{A}} b$ if $\Omega(a) < \Omega(b)$, and $a \sqsubseteq_{\mathbb{A}} b$ if $\Omega(a) \leq \Omega(b)$.

An *alternating* Ω*-chain* of *lenth* k in $\mathbb{A}$ is a sequence $a_0 a_1 \cdots a_k$ of states that all belong to the same cluster and satisfy, for all $i < k$, that $\Omega(a_i) < \Omega(a_{i+1})$ while a_i and a_{i+1} have different parity. $\lhd$

The following proposition is immediate by the definitions.

Proposition 7.9 *Let $\mathbb{A} = \langle A, \Theta, \Omega, a_I\rangle$ and $\mathbb{A}' = \langle A, \Theta', \Omega, a_I\rangle$ be two modal automata such that $\Theta(a,c) \equiv \Theta'(a,c)$ for each $a \in A$ and $c \in \wp(\mathsf{P})$. Then $\mathbb{A} \equiv \mathbb{A}'$.*

Remark 7.10 Another way of defining the semantics of modal automata is via the 'slow' acceptance game of Table 10, which is perhaps closer to the evaluation games of the modal μ-calculus. In this set-up, at a basic position (a, s) $\exists$ does not have to come up with a marking m, but rather, the state a is 'unfolded' into the formula $\Theta(a, \sigma_V(s))$, and the two players engage in a little sub-game in order to determine whether $\Theta(a, \sigma_V(s))$ is true at s or not. At the end of this sub-game, unless one of the players got stuck, the match arrives at another basic position. We leave it as an exercise for the reader to check that the two games are in fact equivalent. $\lhd$

Position	Player	Admissible moves	Priority
$(a, s) \in A \times S$	$-$	$\{(\Theta(a, \sigma_V(s)), s)\}$	$\Omega(a)$
$(\top, s)$	$\forall$	$\varnothing$	0
$(\bot, s)$	$\exists$	$\varnothing$	0
$(\Diamond\pi, s)$	$\exists$	$\{(\pi, t) \mid t \in R(s)\}$	0
$(\Box\pi, s)$	$\forall$	$\{(\pi, t) \mid t \in R(s)\}$	0
$(\varphi_0 \lor \varphi_1, s)$	$\exists$	$\{(\varphi_0, s), (\varphi_1, s)\}$	0
$(\varphi_0 \land \varphi_1, s)$	$\forall$	$\{(\varphi_0, s), (\varphi_1, s)\}$	0

Table 10: Slow acceptance game for modal automata

Regarding complexity matters, we define the *size* of a modal automaton to get a nice fit with the (slow) acceptance game defined in Remark 7.10. In particular, this means that we cannot simply define the size of an automaton as its number of states, we have to take the *transition map* of the device into account as well. Note that the size $|\alpha|$ of a modal one-step α is simply defined as its number of subformulas, or, equivalently, as the size of its closure. The *index* of modal automata is defined in the same way as for parity formulas.

Definition 7.11 Let $\mathbb{A} = (A, \Theta, \Omega, a_I)$ be a modal automaton. The *size* $|\mathbb{A}|$ of $\mathbb{A}$ is defined as follows:

$$|\mathbb{A}| := \sum_{(a,c) \in A \times C} |\Theta(a, c)|.$$

Its *index* $ind(\mathbb{A})$ is given as the maximal length of an alternating Ω-chain in $\mathbb{A}$. $\lhd$

Later on this chapter we will provide effective translations transforming a μ-calculus formula into an equivalent modal automaton, and vice versa. As a corollary of this result we obtain that modal automata are bisimulation invariant — in Exercise 7.2 the reader is asked to give a direct proof.

Theorem 7.12 *Let $\mathbb{A}$ be a modal automaton.. Then for any bisimilar pair $(\mathbb{S}, s)$ and $(\mathbb{S}', s')$ of pointed Kripke models it holds that*

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

7.3 Disjunctive modal automata

A key tool in the study of the model μ-calculus is provided by the automata that we are about to introduce now, viz., the nondeterministic variants of the modal automata that we just met in section 7.2. The *disjunctive* automata, as we shall call them, are obtained by restricting the co-domain of the transition map of a modal automaton to the set of so-called disjunctive one-step formulas, which are based on the cover modality discussed in section 1.7.

Definition 7.13 Given a finite set A, we define the set $\mathtt{1DML}(A)$ of disjunctive modal one-step formulas in A as follows

$$\alpha \ ::= \ \bot \ \mid \ \top \ \mid \ \nabla B \ \mid \ \alpha \vee \alpha,$$

where $B \subseteq A$.

A modal P-automaton $\mathbb{A} = (A, \Theta, \Omega, a_I)$ is called *disjunctive* or *non-deterministic* if $\Theta(a, c) \in \mathtt{1DML}(A)$, for every $a \in A$ and $c \in \wp(\mathsf{P})$. ◁

► `example(s) to be supplied`

Remark 7.14 As a variant of Definition 7.13, we will sometimes require that the range of the transition map Θ of a disjunctive automaton is given by the formulas of the slightly more restricted one-step language $\mathtt{1DML}_r$ given by the following grammar:

$$\alpha \ ::= \ \bot \ \mid \ \nabla B \ \mid \ \alpha \vee \alpha,$$

where $B \subseteq A$. In other words, in this set-up every formula $\Theta(a, c)$ is a finite disjunction of nabla formulas; the difference with the language of Definition 7.13 is that here, the formula $\top$ is not allowed.

We leave it as an exercise to the reader to prove that the two versions of the definition are equivalent, in the sense that there are transformations from one type of automaton into the other. ◁

As already mentioned, the key property making an automaton *non-deterministic* is that, on Kripke structure with a sufficiently nice branching structure, a winning strategy for $\exists$ in the acceptance game should always be able to find markings that are *functional*. We will now make this statement more precise.

Definition 7.15 Let $\mathbb{A}$ and $\mathbb{S}$ be a modal automaton and a Kripke structure, respectively. A strategy f for $\exists$ in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ is called *separating* if for all partial matches Σ ending in a basic position (a, s), the marking $m_\Sigma : S \to \wp A$ picked by f satisfies $|m_\Sigma(t)| \leq 1$ for all $t \in S$, and $|m_\Sigma(t)| = 0$ for all $t \notin \sigma_R(s)$. ◁

In words, a strategy is separating if it picks markings that assign to each point in S at most one state in A, and assign the empty set to any point that is not a successor of the currently inspected point of S. For a (non-)example, consider the one-step formula $\Diamond a_0 \wedge \Box a_1$; it should be clear that to satisfy this formula at a point s, one needs at least one successor of s where *both* a_0 and a_1 hold. This means that no separating strategy will prescribe a legitimate move for a position of the form (a, s) if the formula that $\exists$ needs to satisfy is $\Theta(a, \sigma_V(s)) = \Diamond a_0 \wedge \Box a_1$.

Separating winning strategies have the following property, which we will put to good use in the sequel.

Definition 7.16 Let $\mathbb{A}$ and $(\mathbb{S}, r)$ be a modal automaton and a pointed Kripke structure, respectively. A strategy f for $\exists$ in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, r)$ is called *functional* if for every $s \in S$ there is at most one $a \in A$ such that the position (a, s) is reachable in an f-guided match of $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, r)$.

In case $\exists$ has a *functional* winning strategy in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, r)$, we say that $\mathbb{A}$ *strongly* accepts $(\mathbb{S}, r)$, and write $\mathbb{S}, r \Vdash_s \mathbb{A}$. $\lhd$

Proposition 7.17 *Let $\mathbb{A}$ be a modal automaton, and let $(\mathbb{S}, s)$ be a pointed tree model. Then every separating winning strategy in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, s)$ is functional.*

We have now arrived at the key result about disjunctive automata.

Theorem 7.18 *Let $\mathbb{A}$ and $(\mathbb{S}, r)$ be a disjunctive modal automaton and a pointed Kripke model, respectively. Then $\mathbb{S}, r \Vdash \mathbb{A}$ iff there is a rooted tree model $(\mathbb{S}', r')$ such that $\mathbb{S}, r \leftrightarroweq (\mathbb{S}', r')$ and $\mathbb{S}', r' \Vdash_s \mathbb{A}$.*

Proof of Theorem 7.18. With $\mathbb{A} = (A, \Theta, \Omega, a_I)$, let $\kappa := |A|$ be the state-size of $\mathbb{A}$. We leave it for the reader to construct a tree model $\mathbb{S}'$ with root r', and a bounded morphism $g : \mathbb{S}' \to \mathbb{S}$ such that $g(r') = r$ and such that every $s' \neq r'$ in $\mathbb{S}'$ has at least $\kappa - 1$ many siblings t' such that $g(t') = g(s')$.

By positional determinacy we may assume that $\exists$ has a positional strategy f in $\mathcal{A}(\mathbb{A}, \mathbb{S})$ which is winning when played from any winning position for $\exists$. We will use this strategy to define a separating positional winning strategy for $\exists$ in $\mathcal{A}(\mathbb{A}, \mathbb{S}')$.

The key claim is the following.

Claim 1 Let $s \in S$ and $s' \in S'$ be such that $g(s') = s$, let $\alpha \in \mathtt{1DML}(A)$ be a one-step formula and let $m : R(s) \to \wp(A)$ be a marking such that $\mathbb{S} \oplus m, g(s') \Vdash \alpha$. Then there is a separating marking $m' : R'(s') \to \wp(A)$ such that $\mathbb{S}', s' \Vdash \alpha$ and $m'(t') \subseteq m(g(t'))$, for all $t' \in R'(s')$.

Proof of Claim In case α contains $\top$ as one of its disjuncts, we simply take the empty marking for m', that is, we define $m'(t') := \varnothing$ for every $t' \in S'$.

In the sequel we focus on the case where α does not contain $\top$ as one of its disjuncts (in fact this is without loss of generality, cf. Remark 7.14). It follows from the legitimacy of m, as a move for $\exists$ in $\mathcal{A}(\mathbb{A}, \mathbb{S})$, that $\mathbb{S}, m, s \Vdash \alpha$; this means that $\mathbb{S} \oplus m, s \Vdash \nabla B$ for some disjunct ∇B of α, where $B \subseteq A$. We now consider two subcases.

If $B = \varnothing$, it follows from $\mathbb{S} \oplus m, s \Vdash \nabla B$ that $\sigma_R = \varnothing$; but then we also have $\sigma_{R'}(s') = \varnothing$, since g is a bounded morphism. In this case we also define m' as the empty marking.

Finally, assume that $B \neq \varnothing$; since $\mathbb{S} \oplus m, s \Vdash \nabla B$ we may without loss of generality assume that $\varnothing \neq m(t) \subseteq B$, for all $t \in \sigma_R(s)$. Now consider an arbitrary successor t of s. By the assumption on g there are at least κ many successors t' of s' such that $g(t') = t$, and since

$\kappa \geq |\mathbb{A}|$ this implies that there is a surjection $h : g^{-1}(t) \to m(t)$. Define $m' : \sigma_{R'}(s') \to \wp A$ by putting

$$m'(t') := \{h(t')\}.$$

We leave it as an exercise for the reader to check that $\mathbb{S}', m', s' \Vdash \nabla B$. This means that $\mathbb{S}', m', s' \Vdash \alpha$, thus establishing that m' is a legitimate move for $\exists$ at position (a, s') in $\mathcal{A}(\mathbb{A}, \mathbb{S}')$ indeed. Finally, it is immediate from the definition of m' that $m'(t') \subseteq m(g(t'))$, for all $t' \in \sigma_{R'}(s')$. ◀

Based on Claim 1, we may provide $\exists$ with the following positional strategy f' in $\mathcal{A}(\mathbb{A}, \mathbb{S}')$. Given a position (a, s'), in case $(a, g(s'))$ is a winning position for $\exists$ in $\mathcal{A}(\mathbb{A}, \mathbb{S})$, we let f' pick a marking m' as given by the claim, while f' picks an random move in case $(a, g(s')) \notin \mathrm{Win}_{\exists}(\mathcal{A}(\mathbb{A}, \mathbb{S}))$.

It is not hard to prove that for any f'-guided (partial) match $\Sigma = (a_n, s'_n)_{n<\lambda}$ of $\mathcal{A}(\mathbb{A}, \mathbb{S}')$, its g-projection $\Sigma^g := (a_n, g(s'_n))_{n<\lambda}$ is a f-guided (partial) match of $\mathcal{A}(\mathbb{A}, \mathbb{S}')$. From this it is immediate that f' is a winning strategy when played from a winning position, while it is obvious from its definition that f is separating. QED

Further on in this chapter we will prove a *Simulation Theorem*, providing a construction which effectively transforms a given modal automaton into an equivalent disjunctive modal automaton.

7.4 One-step logics and their automata

Modal one-step logic

As we saw in section 7.2, modal one-step formulas provide the co-domain of the transition map of a modal automaton. The operational semantics of modal automata is given by a two-player acceptance game, and a match of this game proceeds in rounds, during which the players investigate a local window into the Kripke structure, by means of the semantics of one of these one-step formulas. It will be rewarding to introduce some terminology for this 'local window' and study the semantics of one-step formulas in some more detail. This will allow us to introduce the notion of a *one-step logic* and use it to generalise the notion of a modal automaton.

The crucial observation is the following. Consider a modal automaton $\mathbb{A} = (A, \Theta, \Omega, a_I)$ and a Kripke model $\mathbb{S}$. At a basic position (a, s) of the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$, $\exists$ has to come up with a marking m which makes the one-step formula $\Theta(a, \sigma_V)$ true at s in the expanded model $\mathbb{S} \oplus m$. The point is that, because of the special shape of modal one-step formulas, we do not use all information on the model $\mathbb{S} \oplus m$: in fact all we need access to is the set $R[s]$ of successors of s, and the marking m. In the sequel it will convenient to present this information in the format of a *one-step model*, which is nothing but a set, together with a marking for the set of variables.

Definition 7.19 Fix a set A. A *one-step A-model over a set Y* is a pair (Y, m) such that $m : Y \to \wp(A)$ is an *A-marking* of the elements of Y with A-colors. ◁

Remark 7.20 In order to deal with blind worlds (points in a Kripke model that have no successors), we need to allow one-step models with an *empty domain.* Observe that there is in fact exactly one such structure: the pair $(\varnothing, \varnothing)$. Apart from this exception, a one-step model is nothing but a *structure* in the sense of first-order model theory, for the signature consisting of a monadic predicate for each element of A. That is, we may consider the A-model (Y, m) as the structure (Y, V_m), simply by representing the marking m by its associated valuation V_m interpreting the variables as subsets of the domain Y. $\lhd$

Definition 7.21 The *one-step satisfaction relation* $\Vdash^1$ between one-step models and modal one-step formulas is defined as follows. Fix a one-step model (Y, m).

First, we define the value $[\![\pi]\!]^0$ of a formula $\pi \in \mathtt{Latt}(A)$ by the following induction:

$$\begin{array}{lcllcl}
[\![a]\!]^0 & := & V_m(a) \quad (= \{t \in Y \mid a \in m(t)\}) & & & \\
[\![\top]\!]^0 & := & Y & [\![\bot]\!]^0 & := & \varnothing \\
[\![\pi_0 \vee \pi_1]\!]^0 & := & [\![\pi_0]\!]^0 \cup [\![\pi_1]\!]^0 & [\![\pi_0 \wedge \pi_1]\!]^0 & := & [\![\pi_0]\!]^0 \cap [\![\pi_1]\!]^0.
\end{array}$$

Sometimes we write $(Y, m), t \Vdash^0 \pi$ in case $t \in [\![\pi]\!]^0$.

Second, we inductively define the one-step satisfaction relation as follows:

$$\begin{array}{lll}
(Y, m) \Vdash^1 \top & & \\
(Y, m) \not\Vdash^1 \bot & & \\
(Y, m) \Vdash^1 \Box\pi & \text{if} & [\![\pi]\!]^0 = Y \\
(Y, m) \Vdash^1 \Diamond\pi & \text{if} & [\![\pi]\!]^0 \cap Y \neq \varnothing \\
(Y, m) \Vdash^1 \alpha_0 \wedge \alpha_1 & \text{if} & (Y, m) \Vdash^1 \alpha_0 \text{ and } (Y, m) \Vdash^1 \alpha_1 \\
(Y, m) \Vdash^1 \alpha_0 \vee \alpha_1 & \text{if} & (Y, m) \Vdash^1 \alpha_0 \text{ or } (Y, m) \Vdash^1 \alpha_1
\end{array}$$

In case $(Y, m) \Vdash^1 \alpha$ we say that α is *true* in the one-step model (Y, m). $\lhd$

Example 7.22 In this format, the semantics of disjunctive formulas boils down to the following, as can easily be verified, for a subset $B \subseteq A$:

$$(Y, m) \Vdash^1 \nabla B \text{ iff } B \subseteq \bigcup\{m(y) \mid y \in Y\} \text{ and } m(y) \cap B \neq \varnothing, \text{ for all } y \in Y.$$

That is, ∇B holds in a one-step model (Y, m) iff every $b \in B$ is satisfied at some $y \in Y$, and every $y \in Y$ satisfies some $b \in B$.

Furthermore, observe that the empty model will satisfy every formula of the form $\Box\pi$, and no formula of the form $\Diamond\pi$. We have $(Y, m) \Vdash^1 \nabla\varnothing$ iff $Y = \varnothing$. $\lhd$

The following proposition, which can be proved by a straightforward induction on the complexity of one-step formulas, shows that the one-step semantics developed above is just an alternative perspective on the standard semantics of one-step formulas.

Proposition 7.23 *Let $\mathbb{S} = (S, R, V)$ be a Kripke model, let s be a point in S, let $m : R[s] \to \wp(A)$ be an A-marking, and let $\alpha \in \mathtt{1ML}(A)$ be a modal one-step formula. Then*

$$\mathbb{S} \oplus m, s \Vdash \alpha \textit{ iff } (R[s], m) \Vdash^1 \alpha.$$

Given Proposition 7.23, the acceptance game of modal automata can now be naturally defined in terms of this one-step semantics, as in Table 11.

Position	Player	Admissible moves	Priority
$(a,s) \in A \times S$	$\exists$	$\{m : R(s) \to \wp A \mid (R(s), m) \Vdash^1 \Theta(a, \sigma_V(s))\}$	$\Omega(a)$
m	$\forall$	$\{(b,t) \mid b \in m(t)\}$	0

Table 11: Acceptance game for one-step automata

General one-step logic

As we will see below, the notion of a one-step logic provides a way to generalise the concept of a modal automaton to a much wider setting.

Definition 7.24 A *one-step language* is a map $\mathtt{L}$ which assigns to any finite set A a collection $\mathtt{L}(A)$ of *one-step formulas over* A. This map is subject to the constraint that every map $\tau : A \to A'$ induces a *substitution* or *renaming* $[\tau] : \mathtt{L}(A) \to \mathtt{L}(A')$ such that
1) $[id_A] = id_{\mathtt{L}(A)}$;
2) $[\tau' \circ \tau] = [\tau'] \circ [\tau]$, for any pair $\tau : A \to A'$ and $\tau' : A' \to A''$;
3) $\alpha[\tau] = \alpha$ for any $\alpha \in \mathtt{L}(A)$, if $\tau : A \to A'$ is such that $\tau(a) = a$ for all $a \in A$. $\lhd$

We will use postfix notation for this renaming, writing $\alpha[\tau]$ for the formula we obtain from α by renaming every variable $a \in A$ by $\tau(a) \in A'$. For instance, where $\alpha \in \mathtt{1ML}(A)$ is the formula $\Diamond a \wedge \Box(b \vee c)$ and $\tau : A \to A'$ satisfies $\tau(a) = \tau(c) = a'$ and $\tau(b) = b'$, we find $\alpha[\tau] = \Diamond a' \wedge \Box(b' \vee a')$. Note that it follows from the above definition that $A \subseteq A'$ implies $\mathtt{L}(A) \subseteq \mathtt{L}(A')$, for any one-step language $\mathtt{L}$.

Definition 7.25 A *one-step logic* is a pair $(\mathtt{L}, \Vdash^1)$ consisting of a one-step language $\mathtt{L}$ and an *interpretation* $\Vdash^1$ which indicates, for every one-step A-model (Y, m) and every one-step formula $\alpha \in \mathtt{L}(A)$, whether α is *true* or *false* in (Y, m), denoted as, respectively, $(Y, m) \Vdash^1 \alpha$ and $(Y, m) \not\Vdash^1 \alpha$.

The interpretation $\Vdash^1$ is subject to the condition of *monotonicity*: if $m(t) \subseteq m'(t)$, for all $t \in Y$, then $(Y, m) \Vdash^1 \alpha$ implies $(Y, m') \Vdash^1 \alpha$, for all $\alpha \in \mathtt{L}(A)$. Furthermore, the interpretation is supposed to be well-behaved with respect to renamings, in the following sense. Observe that a map $\tau : A' \to A$ transforms any A-valuation $V : A \to \wp(Y)$ to an A'-valuation $V \circ \tau : A' \to \wp(Y)$; we will require that $(Y, m_V) \Vdash^1 \alpha[\tau]$ iff $(Y, m_{V\circ\tau}) \Vdash^1 \alpha$, for any formula $\alpha \in \mathtt{L}(A)$. $\lhd$

We will generally be sloppy and blur the distinction between a one-step language and a one-step logic, in the understanding that the interpretation of one-step languages is generally fixed (and always clear from context).

In Definition 7.21 we introduced the one-step perspective on modal logic. As a different, particularly interesting example of a one-step logic, we may consider two versions of *monadic first-order logic*, where we see the variables in A as monadic predicate symbols.

Definition 7.26 The set $\mathtt{MFOE}(A)$ of *monadic first-order formulas over* A is given by the following grammar:

$$\alpha \ ::= \ \top \mid \bot \mid a(x) \mid \neg a(x) \mid x \doteq y \mid x \neq y \mid \alpha \vee \alpha \mid \alpha \wedge \alpha \mid \exists x.\alpha \mid \forall x.\alpha$$

where $a \in A$ and x, y are first-order (individual) variables. The language $\mathtt{MFO}(A)$ of *monadic first-order logic* is the equality-free fragment of $\mathtt{MFOE}(A)$; that is, atomic formulas of the form $x \doteq y$ and $x \not\doteq y$ are not permitted:

$$\alpha \ ::= \ \top \mid \bot \mid a(x) \mid \neg a(x) \mid \alpha \vee \alpha \mid \alpha \wedge \alpha \mid \exists x.\alpha \mid \forall x.\alpha$$

In both languages we use the standard definition of free and bound variables, and we call a formula a *sentence* if it has no free variables. For each of the languages $\mathtt{L} \in \{\mathtt{1FO}, \mathtt{1FOE}\}$, we define the *positive fragment* $\mathtt{L}^+$ of $\mathtt{L}$ as the language obtained by almost the same grammar as for $\mathtt{L}$, but with the difference that we do not allow negative formulas of the form $\neg a(x)$ (but do allow formulas $x \not\doteq y$). $\lhd$

To define the *semantics* of these formulas, we make a distinction between the empty one-step model and non-empty models, cf. Remark 7.20. In the latter case we view a one-step model (Y, m) as the first-order structure (Y, V_m). If we add to such a model an *assignment* g, interpreting individual variables of the language as elements of the domain, we may inductively define, in a completely straightforward way, the notion of a monadic formulas being *true* in a model-with-assignment:

$$(Y, m), g \models \alpha.$$

Note the truth of a *sentence* of the language does not depend on the assignment, so that may simply write

$$(Y, m) \models \alpha$$

in case $(Y, m), g \models \alpha$ for some/each assignment.

The *empty* model must be dealt with differently. Since we cannot define assignments on the empty model in a meaningful way, we cannot interpret arbitrary formulas in the empty model. Fortunately, however, we can give an interpretation for every *sentence* of the language, simply by making every formula of the form $\forall x.\alpha$ *true*, and every formula of the form $\exists x.\alpha$ *false* in the empty model. Using this as a basis for an inductive definition, we easily define a truth relation

$$(\varnothing, \varnothing) \models \alpha$$

for any monadic first-order sentence α.

In the light of the above discussion, we will take the (positive) sentences of the languages $\mathtt{MFOE}(A)$ and $\mathtt{MFOE}(A)$ as two respective one-step languages.

Definition 7.27 We define the *one-step languages* $\mathtt{1FOE}(A)$ and $\mathtt{1FO}(A)$ as the collection of *positive sentences* in $\mathtt{MFOE}(A)$ and $\mathtt{MFOE}(A)$, respectively. The semantics $\Vdash^1$ of these languages is defined by putting

$$(Y, m) \Vdash^1 \alpha \text{ iff } (Y, m) \models \alpha,$$

for any one-step model (Y, m). $\lhd$

One-step logic

Continuing our general discussion, we introduce some natural notions pertaining to one-step logics.

Definition 7.28 Two one-step formulas α and α' are *(one-step) equivalent*, denoted $\alpha \equiv_1 \alpha'$, if they are satisfied by exactly the same one-step models. $\lhd$

Example 7.29 Examples of one-step equivalent pairs of formulas include instance of the standard propositional distributive laws, such as the modal distributive law:

$$(\Diamond a_1 \vee \Diamond a_2) \wedge \Box b \equiv_1 (\Diamond a_1 \wedge \Box b) \vee (\Diamond a_2 \wedge \Box b),$$

the familiar axioms of modal logic, such as

$$\Box(a \wedge b) \equiv_1 \Box a \wedge \Box b,$$

but also formulas involving the nabla modality, such as

$$\nabla B \wedge \nabla B' \equiv_1 \bigvee \Big\{ \nabla\{b \wedge b' \mid bRb'\} \mid R \subseteq B \times B' \text{ and } (B, B') \in \overline{\wp}R \Big\}$$

(cf. Proposition 1.34(1)).

Examples such as

$$\Diamond(a_1 \wedge a_2) \wedge \Box b \equiv_1 \exists x\, (a_1(x) \wedge a_2(x)) \wedge \forall y\, b(y).$$

show that Definition 7.28 also covers the notion of one-step equivalence across languages. $\lhd$

We may lift the notion of equivalence to the level of one-step logics.

Definition 7.30 We say that two one-step $(\mathtt{L}, \Vdash^1)$ and $(\mathtt{L}', \Vdash^{1\prime})$ languages are *(effectively) equivalent* if for every formula in $\mathtt{L}$ there is an (effectively obtainable) equivalent formula in $\mathtt{L}'$, and vice versa. $\lhd$

A particular interesting example of such an equivalence is the following.

Proposition 7.31 *The one-step languages* $\mathtt{1ML}$ *and* $\mathtt{1FO}$ *are effectively equivalent.*

Proof. It is easy to rewrite a modal one-step formula into an equivalent first-order formula. For the opposite direction, the key observation is that in equality-free *monadic* first-order logic, every formula can be rewritten into a normal form where every monadic predicate is in the scope of exactly one quantifier. QED

Among the results about the modal one-step language that we shall need later is the following one-step version of the usual bisimulation invariance result for modal logic, i.e. all one-step formulas are invariant for bisimulations between one-step models in a precise sense.

Definition 7.32 We say that two one-step A-models (Y, m) and (Y', m') are *one-step bisimilar*, notation: $(Y, m) \leftrightarroweq^1 (Y', m')$, if they satisfy the following conditions:

(forth) for all $s \in S$, there is $s' \in S'$ with $m(s) = m'(s')$;

(back) for all $s' \in S'$, there is $s \in S$ with $m(s) = m'(s')$. $\triangleleft$

Proposition 7.33 (One-step Bisimulation Invariance) *Let (Y, m) and (Y', m') be two one-step A-models. If $(Y, m) \leftrightarroweq^1 (Y', m')$, then both one-step models satisfy the same formulas in $\mathtt{1ML}(A)$.*

Automata for one-step logics

We now see how the concept of one-step logic naturally give rise to the following generalisation of modal automata.

Definition 7.34 Let $(\mathtt{L}, \Vdash^1)$ be a one-step logic. An $\mathtt{L}$-*automaton* over a set P of proposition letters is a quadruple $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$, where A is a finite state set with initial state a_I, $\Theta : A \times \wp(\mathsf{P}) \to \mathtt{L}(A)$ is a transition function, and $\Omega : A \to \omega$ is a priority map.

The *semantics* of $\mathtt{L}$-automata is given by a two-player acceptance game, of which the rules are given in exactly the same way as those for modal automata, cf. Table 11. $\triangleleft$

As we will see later on, the automata for $\mathtt{1FO}$ and $\mathtt{1FOE}$ are of particular interest since they correspond to, respectively, the modal μ-calculus and (on tree models) monadic second-order logic. The first observation is immediate by our earlier observations on the equivalence of $\mu\mathtt{ML}$ and modal automata, and Proposition 7.31.

An important theme in the study of these automata is how their properties are already determined at the one-step level. Here are some first examples, regarding the closure properties of $\mathtt{L}$-automata. Recall that a *query* is simply a class of pointed Kripke models.

Definition 7.35 Given be a one-step logic $(\mathtt{L}, \Vdash^1)$, we call a query K $\mathtt{L}$-*recognisable* if there is some $\mathtt{L}$-automaton $\mathbb{A}$ that *recognises* K, i.e., such that $\mathbb{S}, s \Vdash \mathbb{A}$ iff $\mathbb{S}, s$ belongs to K. $\triangleleft$

We will generally be interested in *closure properties* of the class of recognisable queries. It is rather easy to see that if a one-step language is closed under taking conjunctions/disjunctions, then the associated class of recognisable languages is closed under taking intersections/unions. The question of closure under complementation is more interesting; note that since our one-step languages consist of *monotone* formulas only, closure under negation at the one-step level is not possible.

Definition 7.36 Let $(\mathtt{L}, \Vdash^1)$ be a one-step logic. We say that $\mathtt{L}$ is *closed under taking conjunctions*, if, given a pair of one-step formulas α and β, there is a one-step formula γ such that any one-step model satisfies γ iff it satisfies both α and β. The notion of *closure under disjunctions* is defined analogously.

Given two one-step formulas α and β in $\mathtt{L}(A)$, we call β a *boolean dual* of α if for every one-step model (Y, m) we have that

$$(Y, m) \Vdash^1 \beta \text{ iff } (Y, \overline{m}) \not\Vdash^1 \alpha,$$

where $\overline{m}$ is the *complement marking* of m, given by $\overline{m}(t) := A \setminus m(t)$, for all $t \in Y$. We say that $\mathtt{L}$ is *closed under taking boolean duals* if every formula in $\mathtt{L}$ has a boolean dual in $\mathtt{L}$. $\lhd$

Example 7.37 The one-step modal language is closed under taking conjunctions, disjunctions and boolean duals. We let α^∂ be the formula we obtain from a formula $\alpha \in \mathtt{1ML}$ by simultaneously replacing all occurrences of $\bot$ by $\top$, all conjunctions by disjunctions, all diamonds by boxes, and vice versa. For example: $\big(\Diamond\top \wedge \Box(a \vee b)\big)^\partial = \Box\bot \vee \Diamond(a \wedge b)$. It is easy to verify that for every $\alpha \in \mathtt{1ML}$, the formulas α and α^∂ are boolean duals of one another.

The one-step language of *disjunctive* modal logic is closed under taking disjunctions, but not conjunctions or boolean duals. $\lhd$

Proposition 7.38 *Let* $(\mathtt{L}, \Vdash^1)$ *be a one-step logic.*

1) If $\mathtt{L}$ *is closed under taking conjunctions, then the* $\mathtt{L}$*-recognisable queries are closed under taking intersections.*

2) If $\mathtt{L}$ *is closed under taking disjunctions, then the* $\mathtt{L}$*-recognisable queries are closed under taking unions.*

3) If $\mathtt{L}$ *is closed under taking boolean duals, then the* $\mathtt{L}$*-recognisable queries are closed under complementation.*

Proof. We leave the proof of the first two statements as an exercise to the reader. For the proof of the third part we need to show that with any $\mathtt{L}$-automaton $\mathbb{A}$ we can associate an $\mathtt{L}$-automaton $\overline{\mathbb{A}}$ which accepts exactly those pointed Kripke models that are rejected by $\mathbb{A}$.

Let $\mathbb{A} = (A, \Theta, \Omega, a_I)$ be an $\mathtt{L}$-automaton, and define $\overline{\mathbb{A}}$ to be the structure $\overline{\mathbb{A}} := (A, \Theta^\partial, \Omega', a_I)$ given by putting $\Theta^\partial(a, c) := \Theta(a, c)^\partial$ and $\Omega'(a) := 1 + \Omega(a)$.

Now take an arbitrary pointed Kripke model $(\mathbb{S}, s)$. Comparing the acceptance games $\mathcal{A}(\mathbb{A}, \mathbb{S})$ and $\mathcal{A}(\overline{\mathbb{A}}, \mathbb{S})$ we observe that the role of $\exists$ in the latter game is basically the same as that of $\forall$ in the first. From this it follows that any position (a, s) is winning for $\exists$ in $\mathcal{A}(\overline{\mathbb{A}}, \mathbb{S})$ iff it is winning for $\forall$ in $\mathcal{A}(\mathbb{A}, \mathbb{S})$. Using determinacy we derive that $\mathbb{S}, s \Vdash \overline{\mathbb{A}}$ iff $\mathbb{S}, s \not\Vdash \mathbb{A}$, as required. QED

7.5 From formulas to automata and back

In this section we will substantiate our earlier claim that modal automata are indeed an alternative way to look at the modal μ-calculus. That is, we will provide effective constructions that transform a (parity) formula into an equivalent modal automaton, and vice versa. In both directions we will let these transformations pass via the intermediate structures of *transparent modal automata*; these are variations of modal automata in which the proposition letters, instead of featuring as part of the domain of the transition map, may occur on the co-domain side. That is, we have to extend the definition of one-step formulas, allowing (unguarded) occurrences of proposition letters.

Definition 7.39 Given a set P of proposition letters and a set A of propositional variables, we define the set $\mathtt{1EML}(\mathsf{P}, A)$ of *extended* one-step modal formulas over P and A using the following grammar:

$$\alpha ::= \bot \mid \top \mid p \mid \overline{p} \mid \Diamond\pi \mid \Box\pi \mid \alpha \wedge \alpha \mid \alpha \vee \alpha,$$

with $P \in \mathsf{P}$ and $\pi \in \mathtt{Latt}(A)$. $\lhd$

Observe that in an extended modal one-step formula, the proposition letters from P may only occur 'at the surface', that is, *not* in the scope of a modality; as in $\mathtt{1ML}(A)$-formulas, every occurrence of a variable from A must be in the scope of exactly one modality.

Definition 7.40 A *transparent modal automaton* over a set P of proposition letters is a quadruple of the form $\mathbb{A} = (A, \Theta, \Omega, a_I)$, where A is a finite set of states, of which a_I is the *initial* state, $\Omega : A \to \omega$ is a priority map, and

$$\Theta : A \to \mathtt{1EML}(\mathsf{P}, A)$$

is the transition map.

Given a Kripke model $\mathbb{S} = (S, R, V)$, we define the *acceptance game* $\mathcal{A}(\mathbb{A}, \mathbb{S})$ as the parity game of which the admissible moves and the priority map are given in Table 12. $\lhd$

Position	Player	Admissible moves	Priority
$(a, s) \in A \times S$	$-$	$\{(\Theta(a), s)\}$	$\Omega(a)$
(p, s), with $p \in \mathsf{P}$ and $s \in V(p)$	$\forall$	$\varnothing$	0
(p, s), with $p \in \mathsf{P}$ and $s \notin V(p)$	$\exists$	$\varnothing$	0
$(\overline{p}, s)$, with $p \in \mathsf{P}$ and $s \in V(p)$	$\exists$	$\varnothing$	0
$(\overline{p}, s)$, with $p \in \mathsf{P}$ and $s \notin V(p)$	$\forall$	$\varnothing$	0
$(\top, s)$	$\forall$	$\varnothing$	0
$(\bot, s)$	$\exists$	$\varnothing$	0
$(\varphi_0 \vee \varphi_1, s)$	$\exists$	$\{(\varphi_0, s), (\varphi_1, s)\}$	0
$(\varphi_0 \wedge \varphi_1, s)$	$\forall$	$\{(\varphi_0, s), (\varphi_1, s)\}$	0
$(\Diamond\pi, s)$	$\exists$	$\{(\pi, t) \mid t \in R(s)\}$	0
$(\Box\pi, s)$	$\forall$	$\{(\pi, t) \mid t \in R(s)\}$	0

Table 12: Acceptance game for transparent modal automata

The key feature of this acceptance game is that at a basic position of the form $(a, s) \in A \times S$, the one-step formula $\Theta(a)$ that $\exists$ needs to satisfy at s does not depend on the colour of s. On the other hand, this formula may now contain literals over P, and in this way the colour of s does play a role when the players evaluate the truth of $\Theta(a)$.

In the sequel we will refer to standard modal automata (i.e., as given in Definition 7.3) as *chromatic* to distinguish them from the transparent ones introduced here.

The main part of this section consists of constructions that transform chromatic modal automata into transparent ones and vice versa, and transform parity formulas into transparent modal automata and vice versa. In all cases we will compare the *size* and *index* of the input and the output structure (these notions are defined for transparent automata as for chromatic ones). Throughout the remainder we fix a set P of proposition letters, and we think of the sizes of P and $\wp(\mathsf{P})$ as being constant.

Proposition 7.41 *There is an effective construction that transforms a transparent modal* P*-automaton* $\mathbb{A}$ *into a chromatic modal* P*-automaton* $\mathbb{A}^c$*, such that*

1) $\mathbb{A}^c \equiv \mathbb{A}$;
2) $|\mathbb{A}^c| = \mathcal{O}(|\mathbb{A}|)$;
3) $ind(\mathbb{A}^c) = ind(\mathbb{A})$.

Proof. The intuition behind the transformation is that in the acceptance game for a transparent automaton we may encounter literals over P, which are to be evaluated at the current state. Depending on the colour of the current state, every such literal will be evaluated to be either true or false. This means, that if we fix this colour, as we do in the acceptance game of a chromatic automaton, we can simply *replace* every literal with the appropriate boolean constant ($\top$ or $\bot$), thus obtaining at a one-step formula in the 'not-extended' language $\mathtt{1ML}(A)$. Performing this substitution systematically, we arrive at the following definitions.

Given a colour $c \in \wp(\mathsf{P})$, we define the *substitution* $\tau_c : \mathtt{1EML}(\mathsf{P}, A) \to \mathtt{1ML}(A)$ given by

$$\tau_c(p) := \begin{cases} \top & \text{if } p \in c \\ \bot & \text{if } p \notin c. \end{cases}$$

Based on this we go from a transparent modal automaton $\mathbb{A} = (A, \Theta, \Omega, a_I)$ to its chromatic counterpart $\mathbb{A}^c := (A, \Theta', \Omega, a_I)$ by putting

$$\Theta'(a, c) := \Theta(a)[\tau_c].$$

The key observation about these substitutions is that for any Kripke model $\mathbb{S} = (S, R, V)$ over P, any s in $\mathbb{S}$, any A-marking m on s, and any extended one-step formula α we have

$$\mathbb{S} \oplus m, s \Vdash \alpha \text{ iff } \mathbb{S} \oplus m, s \Vdash \alpha[\tau_{c_s}],$$

where c_s is the colour of s under V.

It is this equivalence that enables us to move smoothly between the acceptance games $\mathcal{A}(\mathbb{A}, \mathbb{S})$ and $\mathcal{A}(\mathbb{A}^c, \mathbb{S})$: it shows that at any basic position (a, s), any marking $m : S \to \wp(A)$ is legitimate in $\mathcal{A}(\mathbb{A}, \mathbb{S})$ iff it is legitimate in $\mathcal{A}(\mathbb{A}^c, \mathbb{S})$. From this we easily infer that the winning positions for $\exists$ in the two games coincide, which clearly suffices to prove the equivalence of $\mathbb{A}$ and $\mathbb{A}^c$ (1). The statements (2) and (3) are trivial consequences of the definitions. QED

In the opposite direction there is an equally simple transformation.

Proposition 7.42 *There is an effective construction that transforms a chromatic modal* P*-automaton* $\mathbb{A}$ *into a transparent modal* P*-automaton* $\mathbb{A}^t$*, such that*

1) $\mathbb{A}^t \equiv \mathbb{A}$;
2) $|\mathbb{A}^t| = \mathcal{O}(|\mathbb{A}|)$;
3) $ind(\mathbb{A}^t) = ind(\mathbb{A})$.

Proof. Let $\mathbb{A} = (A, \Theta, \Omega, a_I)$ be a chromatic automaton over some set P of proposition letters. We will define $\mathbb{A}^t := (A, \Theta^t, \Omega, a_I)$, where $\Theta^t : A \to \mathtt{1EML}(\mathsf{P}, A)$ is given by

$$\Theta^t(a) := \bigvee_{c \in \wp(\mathsf{P})} \Big(\odot c \wedge \Theta(a, c) \Big).$$

Here $\odot c$ is the formula 'exactly c':

$$\odot c := \bigwedge_{p \in c} p \wedge \bigwedge_{p \in \mathsf{P} \setminus c} \overline{p},$$

which holds in a state s in a Kripke model over P if c is exactly the colour of s. It is easily verified that $\mathbb{A}^t$ satisfies the conditions listed in the statement of the theorem. QED

We now turn to the equivalence of parity formulas and transparent modal automata. The transformation of the first into the latter type of structure is the most complex construction in this section — but the hardest part of the work has already been done in section 6.4 where we discussed *guarded transformations* of parity formulas.

Proposition 7.43 *There is an effective construction that transforms a parity* P*-formula* $\mathbb{G}$ *into a transparent modal* P*-automaton* $\mathbb{A}_{\mathbb{G}}$*, such that*

1) $\mathbb{A}_{\mathbb{G}} \equiv \mathbb{G}$*;*
2) $|\mathbb{A}_{\mathbb{G}}| \leq 2^{\mathcal{O}(|\mathbb{G}|)}$
3) $ind(\mathbb{A}_{\mathbb{G}}) = ind(\mathbb{G})$*.*

Proof. Recall that by Theorem 6.42 there is an algorithm that transforms $\mathbb{G}$ into an equivalent strongly guarded parity formula $\mathbb{H}$ of size (roughly) exponential in $|\mathbb{G}|$, and index $ind(\mathbb{H}) = ind(\mathbb{G})$. Without loss of generality we may assume that every state of $\mathbb{H}$ is the successor of some modal node, cf. Remark 6.45.

The transparent modal automaton $\mathbb{A}$ will be directly based on $\mathbb{H}$. First of all, we let the carrier A of $\mathbb{A}$ be the set of successors of modal nodes, together with the initital vertex v_I, that is:

$$A := \{v_I\} \cup E[V_m].$$

Clearly then all states of $\mathbb{H}$ belong to A, and with every modal node u we may associate an element $a_u \in A$: its unique successor. We define $a_I := v_I$, and as the priority map of $\mathbb{A}$ we take the map $\Omega' : A \to \omega$ given by

$$\Omega'(a) := \begin{cases} \Omega(a) & \text{if } a \in \mathsf{Dom}(\Omega) \\ 0. & \text{otherwise} \end{cases}$$

It is left to define the transition map $\Theta : A \to \mathtt{1EML}(\mathsf{P}, A)$. Basically, for any $a \in A$ we will read off $\Theta(a)$ from a directed acyclic graph $\mathbb{D}_a := (D_a, E_a)$ that we will cut out from the underlying graph (V, E) of $\mathbb{H}$. We define D_a as the smallest subset D of V that contains a and is closed under taking E-successors of non-modal nodes (that is, if $v \in D \setminus V_n$, then $E[v] \subseteq D$). Clearly, any node $u \in D_a$ must be either modal or atomic if $E[u]$ is empty, and either boolean or silent if it is not. The relation E_a can now be defined as follows:

$$E_a := \{(u, v) \in E \cap (D_a \times D_a) \mid v \neq a\}.$$

It follows from the strong guardedness of $\mathbb{H}$ that $\mathbb{D}$ is *acyclic*, so that we may use the relation E_a for recursive definitions. (It is for this reason that we did not define E_a as the *restriction* of E to the set D_a; this would create cycles in case D_a would contain a modal node u such that Eua.) In particular, we will define a formula $\theta_a(u) \in \mathtt{1EML}$ for every $u \in D_a$:

$$\theta_a(u) := \begin{cases} L(u) & \text{if } u \text{ is atomic} \\ \heartsuit a_u & \text{if } u \text{ is modal and } L(u) = \heartsuit \\ \bigodot\{L(v) \mid Euv\} & \text{if } u \text{ is boolean and } L(u) = \odot \\ \theta_a(v) & \text{if } L(u) = \epsilon \text{ and } Euv. \end{cases}$$

Finally, then, we define

$$\Theta(a) := \theta_a(a).$$

It is easy to verify that every formula of the form $\theta_a(u)$ is an extended modal one-step formula over P and A. This implies that $\Theta : A \to \mathtt{1EML}(\mathsf{P}, A)$ is of the required type.

It is an immediate consequence of the definitions that $|\mathbb{A}| \leq \mathbb{H}$ and $ind(\mathbb{A}) \leq ind(\mathbb{H})$; from this we obtain the items (2) and (3) of the theorem. It thus remains to prove the equivalence of $\mathbb{A}$ and $\mathbb{H}$. But a moment of reflection will show that, for any Kripke model $\mathbb{S}$, the evaluation game $\mathcal{E} := \mathcal{E}(\mathbb{H}, \mathbb{S})$ and the acceptance game $\mathcal{A} := (\mathbb{A}, \mathbb{S})$ are *isomorphic*, apart from the automatic moves of type $(a, s) \to (\Theta(a), s)$ in $\mathcal{A}$, which have no counterpart in $\mathcal{E}$. QED

Proposition 7.44 *There is an effective construction that transforms a transparent modal* P*-automaton* $\mathbb{A}$ *into a parity* P*-formula* $\mathbb{G}_{\mathbb{A}}$*, such that*

1) $\mathbb{G}_{\mathbb{A}} \equiv \mathbb{A}$*;*
2) $|\mathbb{G}_{\mathbb{A}}| = |\mathbb{A}|$*;*
3) $ind(\mathbb{G}_{\mathbb{A}}) = ind(\mathbb{A})$*.*

Proof. Given $\mathbb{A} = (A, \Theta, \Omega, a_I)$, define $\mathbb{G}_{\mathbb{A}} = (V, E, L, \Omega, v_I)$ by putting

$$\begin{array}{lcl} V & := & A \cup \bigcup_{a \in A} Sfor(\Theta(a)) \\ E & := & \big\{(a, \Theta(a)) \mid a \in A\big\} \cup \big(\rhd_0 \cap (V \times V)\big) \\ \Omega(v) & := & \begin{cases} \Omega(v) & \text{if } v \in A \\ \uparrow & \text{otherwise} \end{cases} \\ v_I & := & a_I, \end{array}$$

where we recall that $\rhd_0$ is the converse of the direct subformula relation $\lhd_0$. We leave it for the reader to verify that $\mathbb{G}_{\mathbb{A}}$ satisfies the conditions (1), (2) and (3). QED

7.6 Simulation Theorem

In this section we will prove the most important result of this chapter, viz., the *Simulation Theorem* stating that every modal automaton can be replaced with an equivalent disjunctive modal automaton.

Theorem 7.45 *There is a construction* sim *transforming a modal automaton* $\mathbb{A}$ *into an equivalent disjunctive modal automaton* $\mathsf{sim}(\mathbb{A})$*.*

The definition of the simulating automaton proceeds in two stages. We first come up with an automaton $\mathbb{A}^\sharp$ of which the transition map already has the right shape, but the acceptance condition is not a parity condition but a so-called *ω-regular* set over the carrier $A^\sharp$ of $\mathbb{A}^\sharp$ (i.e., a subset of $(A^\sharp)^\omega$ that itself can be recognized by some finite stream automaton with a parity acceptance condition). As we shall see, the move from $\mathbb{A}$ to $\mathbb{A}^\sharp$ involves a 'change of basis': the states of $\mathbb{A}^\sharp$ will be taken from the set $A^\sharp := \wp(A \times A)$ of binary relations over A, and the definition of the transition map $\Theta^\sharp$ of $\mathbb{A}^\sharp$ is based on various links between the one-step languages we obtain by taking A and $A^\sharp$ as sets of (formal) variables. In the second step of the construction we then show how $\mathbb{A}^\sharp$, like any automaton with an ω-regular acceptance condition, can be transformed into a standard modal automaton with a parity condition.

In fact, we shall prove a slightly more general version of Theorem 7.45, by abstracting from the precise shape of the one-step languages $\mathtt{1ML}$ and $\mathtt{1DML}$ that form the codomain of the transition function of modal and disjunctive modal automata, respectively. Our proof will only use a certain distributive law that holds between $\mathtt{1ML}(A)$ and $\mathtt{1DML}(A)$, and for future reference it will make sense to formulate our definitions and results for two arbitrary one-step languages satisfying such a distributive law.

Convention 7.46 Throughout this section we we shall be dealing with two one-step languages $\mathtt{L}_1$ and $\mathtt{L}_2$, providing sets $\mathtt{L}_i(A)$ of formulas for each set A of propositional variables.

Recall that, in line with the context of fixpoint logics that we are working in, we will assume that, for any one-step logic $\mathtt{L}$, the formulas in $\mathtt{L}(A)$ are all *monotone*. Recall as well that in Definition 7.34 we introduced the notion of an $\mathtt{L}$-automaton, and that in Table 11 we summarize the rules of the acceptance game of such automata.

Our purpose will be to prove that, under some natural constraints on the relation between two one-step languages $\mathtt{L}_1$ and $\mathtt{L}_2$, every $\mathtt{L}_1$-automaton can be *simulated by* an $\mathtt{L}_2$-automaton, that is, transformed into an equivalent $\mathtt{L}_2$-automaton. In the case where $\mathtt{L}_1 = \mathtt{1ML}$ and $\mathtt{L}_2 = \mathtt{1DML}$, the simulating language $\mathtt{1DML}$ corresponds to some *fragment* of $\mathtt{1ML}$, in which the use of conjunctions is severely restricted. Here the construction of the simulating automaton corresponds to finding a *disjunctive normal form* for the modal automata.

In order to formulate the condition on $\mathtt{L}_1$ and $\mathtt{L}_2$ under which we can prove a simulation theorem, we need some preparatory work. Informally, let $\mathtt{L}^\wedge(A)$ denote the version of the language $\mathtt{L}$ that allows conjunctions of proposition letters from A to occur at positions where $\mathtt{L}$ only allows the proposition letters from A themselves. As an example, recall that the language $\mathtt{1DML}(A)$ is built up from basic formulas ∇B, where $B \subseteq A$. Examples of formulas in $\mathtt{1DML}^\wedge(A)$ are $\nabla\{a \wedge b, b\}$ and $\bot \vee \nabla\{a_1 \wedge a_2 \wedge a_3, \top\}$. Observe that these two formulas do not belong to $\mathtt{1DML}(A)$, and thus bear witness to the fact that the latter language forms a *proper* subset of $\mathtt{1DML}^\wedge(A)$. On the other hand, it is easy to see that $\mathtt{1ML}(A) = \mathtt{1ML}^\wedge(A)$.

A convenient way of thinking about the formulas in $\mathtt{L}^\wedge(A)$ is that they are *substitution instances* of formulas in $\mathtt{L}(\wp A)$ under a special substitution θ_A. Formally we define the languare as follows.

Definition 7.47 For any set A and any language $\mathtt{L}$, we define the language

$$\mathtt{L}^\wedge(A) := \{\varphi[\theta_A] \mid \varphi \in \mathtt{L}(\wp A)\},$$

where we let θ_A denote the substitution that replaces, for any subset $B \subseteq A$, the (formal) variable B with the conjunction $\bigwedge B$. $\lhd$

As an example, we obtain the formula $\Box a \wedge \Box(a \wedge b) \in \mathtt{1ML}(A)$ from the formula $\Box\{a\} \wedge \Box\{a, b\} \in \mathtt{1ML}(\mathsf{P}, \wp A)$ by substituting $a = \bigwedge\{a\}$ for $\{a\}$, and $a \wedge b = \bigwedge\{a, b\}$ for $\{a, b\}$.

Now we can define the key condition on two languages $\mathtt{L}_1$ and $\mathtt{L}_2$, making that $\mathtt{L}_2$-automata can simulate $\mathtt{L}_1$-automata, as follows.

Definition 7.48 $\mathtt{L}_2$ is $\bigwedge$-*distributive* over $\mathtt{L}_1$ if, for each set A, and for every finite set Φ of $\mathtt{L}_1(A)$-formulas we have

$$\bigwedge \Phi \;\equiv\; \psi[\theta_A],$$

for some formula $\psi \in \mathtt{L}_2(\wp A)$. $\lhd$

Informally, $\mathtt{L}_2$ is $\bigwedge$-distributive over $\mathtt{L}_1$ if every finite conjunction of $\mathtt{L}_1(A)$-formulas is equivalent to some $\mathtt{L}_2^{\wedge}(A)$-formula. The terminology can be motivated as follows: $\mathtt{L}_2$ is $\bigwedge$-distributive over $\mathtt{L}_1$ if every conjunction of $\mathtt{L}_1$-formulas is equivalent to an $\mathtt{L}_2$-formula of conjunctions; that is, if conjunctions in $\mathtt{L}_1$ 'distribute over $\mathtt{L}_2$-formulas'. As a key example of $\bigwedge$-distributivity we have the following result, which can be proved along the same lines as Proposition 1.34.

Proposition 7.49 $\mathtt{1DML}(A)$ *is* $\bigwedge$-*distributive over* $\mathtt{1ML}(A)$.

The importance of the notion of $\bigwedge$-distributivity lies in the following Theorem, which obviously generalises the simulation theorem for modal automata.

Theorem 7.50 (Simulation Theorem) *Let* $\mathtt{L}_1$ *and* $\mathtt{L}_2$ *be two one-step languages such that* $\mathtt{L}_2$ *is* $\bigwedge$-distributive *over* $\mathtt{L}_1$. *Then there is an effective construction* sim *transforming an* $\mathtt{L}_1$-*automaton* $\mathbb{A}$ *into an equivalent* $\mathtt{L}_2$-*automaton* $\mathsf{sim}(\mathbb{A})$.

We now turn to the definition of the $\mathtt{L}_2$-automaton $\mathbb{A}^\sharp$ that simulates an arbitrary but fixed $\mathtt{L}_1$-automaton $\mathbb{A}$. Note that our prime example concerns a simulation theorem where the transition structure of the simulating automaton is of a significantly simpler nature than that of the simulated one. The intuition underlying the definition of $\mathbb{A}^\sharp$ is that one $\mathbb{A}^\sharp$-match will correspond to a *bundle of several* $\mathbb{A}$*-matches in parallel*, and that to win an $\mathbb{A}^\sharp$-match, $\exists$ has to win *each* of these parallel $\mathbb{A}$-matches. It is thus to be expected that we will obtain $\mathbb{A}^\sharp$ via some kind of power construction on $\mathbb{A}$.

For some more detail, suppose that $\exists$ is faced with a set $\{(a, s) \mid a \in B_s\}$ of positions in some $\mathbb{A}$-acceptance game, for some subset $B_s \subseteq A$ (and one single state s). She could try to respond to all challenges posed by these positions *in one go* by coming up with a single marking $m : R[s] \to \wp A$ such that $(R[s], m) \Vdash^1 \bigwedge\{\Theta(a, c_s) \mid a \in B\}$. Then for each such successor t of s, we can see $B_t = m(t)$ as the set of new challenges that she should take care of at t in parallel. In this way, we may think of a match of the simulating automaton moving in rounds, from one 'macro-position' (B_i, s_i) (corresponding to the set $\{(b, s_i) \mid b \in B_i\}$) to another 'macro-position' (B_{i+1}, s_{i+1}) (corresponding to the set $\{(b, s_{i+1}) \mid b \in B_{i+1}\}$).

This approach would suggest to take $\wp A$ as the carrier set of $\mathbb{A}^\sharp$. However, if we would simply take the states of $\mathbb{A}^\sharp$ to be *macro-states* of $\mathbb{A}$, i.e., subsets of $\mathbb{A}$, we would get into trouble when defining the acceptance condition of $\mathbb{A}$, similar to the problems one encounters when determinizing stream automata. The problem is that from a sequence $B_1B_2B_3\dots$ of subsets of A, representing an $\mathbb{A}^\sharp$-match, we cannot recognize the set of parallel $\mathbb{A}$-matches that this sequence corresponds to. We can take an elegant way out of this problem by defining the carrier set $A^\sharp$ of $\mathbb{A}^\sharp$ to be the set of *binary relations* over A, and to link $A^\sharp$-sequences and A-sequences via the notion of a *trace* through a sequence of binary relations.

Definition 7.51 Fix a set A. We let $A^\sharp$ denote the set of binary relations over A, that is,

$$A^\sharp := \wp(A \times A).$$

Given an infinite word $\rho = R_1R_2R_3\dots$ over the set $A^\sharp$, a *trace* through ρ is either a finite A-word $\alpha = a_0a_1a_2\dots a_k$, or an A-stream $\alpha = a_0a_1a_2\dots$, such that $a_iR_{i+1}a_{i+1}$ for all $i < k$ (respectively, for all $i < \omega$). Finite traces through finite $A^\sharp$-sequences are defined similarly. $\lhd$

The key idea behind the definition of $\mathbb{A}^\sharp$ and the proof of its equivalence to $\mathbb{A}$, is that with each $\mathcal{A}(\mathbb{A}^\sharp, \mathbb{S})$-match with basic positions

$$(R_1, s_1)(R_2, s_2)(R_3, s_3)\dots$$

and each trace $a_0a_1a_2$ through $R_1R_2R_3\dots$ we may associate an $\mathcal{A}(\mathbb{A}, \mathbb{S})$-match with basic positions

$$(a_1, s_1)(a_2, s_2)(a_3, s_3)\dots$$

This explains the winning condition of the automaton $\mathbb{A}^\sharp$: an $A^\sharp$-stream should be winning for $\exists$ if *all* traces through it are winning according to the acceptance condition of $\mathbb{A}$.

Definition 7.52 Relative to a parity condition Ω on A, call an infinite trace $\alpha \in A^\omega$ *bad* if the maximum priority occurring infinitely often on α is an odd number. Let NBT_Ω denote the set of infinite $A^\sharp$-words that contain no bad traces relative to Ω. $\lhd$

Note that the automaton $\mathbb{A}^\sharp$ will be equipped with this set NBT_Ω as its acceptance condition, and while we will be able to establish that $\mathbb{A}^\sharp$ is equivalent to $\mathbb{A}$, NBT_Ω clearly is not a parity condition. This we will take care of in the second part of the construction.

Before giving the formal details, let us first provide some further intuitions behind the definition of $\mathbb{A}^\sharp$. Our starting point is that a state R of $\mathbb{A}^\sharp$ encodes the macro-state $\mathsf{Ran}(R) := \{b \in A \mid (a, b) \in R \text{ for some } a \in A\}$, that is, the range of R. This already suffices to motivate the definition of the initial state of $\mathbb{A}^\sharp$:

$$R_I := \{(a_I, a_I)\}.$$

In order to introduce the definition of $\Theta^\sharp : (A^\sharp \times \wp\mathsf{P}) \to \mathsf{L}_2(A^\sharp)$, consider a model $\mathbb{S}$ and a position of the form (R, s) in the acceptance game $\mathcal{G}^\sharp = \mathcal{A}(\mathbb{A}^\sharp, \mathbb{S})$. Take a state $a \in \mathsf{Ran}(R)$, then at the position (a, s) in the game $\mathcal{G} = \mathcal{A}(\mathbb{A}, \mathbb{S})$, $\exists$ has to come up with a marking $m_{a,s} : R[s] \to \wp(A)$ such that $(R[s], m_{a,s}) \Vdash^1 \Theta(a, c_s)$. Since the position (R, s) encodes the 'macro-position' $\{(a,s) \mid a \in \mathsf{Ran}(R)\}$, we need to consider all of the formulas $\Theta(a, c_s)$ (with $a \in \mathsf{Ran}(R)$) in parallel; this would suggest to consider the conjunction $\bigwedge\{\Theta(a, c_s) \mid a \in \mathsf{Ran}(R)\}$. However, in this conjunction we are no longer able to retrieve the 'origin' of a propositional variable $b \in A$. For this reason we use the following trick. We consider any pair $(a,b) \in A \times A$ as a new propositional variable, representing the variable b tagged with the 'origin' a.

Definition 7.53 Given a language $\mathtt{L}$ and a variable a, let τ_a be the substitution replacing any variable $b \in A$ with the variable $(a,b) \in A \times A$. In words, we say that τ_a *tags* each variable b with a. Given a state a of $\mathbb{A}$ and a color $c \in \wp\mathsf{P}$, let $\Theta^\star(a,c) \in \mathtt{L}_1(A \times A)$ be the formula

$$\Theta^\star(a,c) := \Theta(a,c)[\tau_a],$$

that is, each $b \in A$ occurring in $\Theta(a)$ is replaced with (a,b). $\lhd$

As an example, if $\Theta(a,c) = \Diamond a \wedge \Box b$, then $\Theta^\star(a,c) = \Diamond(a,a) \wedge \Box(a,b)$.

Using this trick we can think of a state $R \in A^\sharp$ unfolding into the formula $\bigwedge\{\Theta^\star(a, c_s) \mid a \in \mathsf{Ran}(R)\} \in \mathtt{L}_1(A \times A)$. Observe that any variable in this formula that is in the scope of a modality, must be of the form $(a,b) \in A \times A$, thus encoding a 'direct meaning' b together with its 'origin' a. Also note that any binary relation $Q \in A^\sharp$ now represents a set of (formal) variables, and so it makes sense to consider for instance the conjunction $\bigwedge Q$.

The following proposition is immediate by the definitions.

Proposition 7.54 *Let $\mathtt{L}_1$ and $\mathtt{L}_2$ be two languages such that $\mathtt{L}_2$ is $\bigwedge$-distributive over $\mathtt{L}_1$, and let A be some set. Then for every finite set Φ of formulas in $\mathtt{L}_1(A \times A)$ there is a formula $\psi \in \mathtt{L}_2(A^\sharp)$ such that*

$$\bigwedge \Phi \;\equiv\; \psi[\theta_{A\times A}], \tag{54}$$

where $\theta_{A\times A}$ is the substitution replacing every relation $Q \subseteq A \times A$ with the conjunction $\bigwedge Q$.

We are now ready for the formal definition of the automaton $\mathbb{A}^\sharp$.

Definition 7.55 Let $\mathtt{L}_1$ and $\mathtt{L}_2$ be two languages such that $\mathtt{L}_2$ is $\bigwedge$-distributive over $\mathtt{L}_1$, and let $\mathbb{A} = \langle A, \Theta, \Omega, a_I\rangle$ be an $\mathtt{L}_1$-automaton. $\mathbb{A}^\sharp$ is given as the $\mathtt{L}_2$-automaton

$$\mathbb{A}^\sharp := \langle A^\sharp, \Theta^\sharp, \mathrm{NBT}_\Omega, R_I\rangle.$$

Here $A^\sharp = \wp(A \times A)$ is the set of binary relations on A, the initial state R_I is the relation $R_I := \{(a_I, a_I)\}$. The transition function $\Theta^\sharp$ is given by fixing, for $\Theta^\sharp(R,c)$, a formula $\psi \in \mathtt{L}_2(A^\sharp)$ satisfying

$$\bigwedge\{\Theta^\star(a,c) \mid a \in \mathsf{Ran}(R)\} \;\equiv\; \psi[\theta_{A\times A}], \tag{55}$$

Finally, the acceptance condition $\mathrm{NBT}_\Omega \subseteq (A^\sharp)^\omega$ is as given in Definition 7.52. $\lhd$

The main technical result of this section concerns the following equivalence.

Proposition 7.56 *Let L_1 and L_2 be two languages such that L_2 is $\bigwedge$-distributive over L_1, and let $\mathbb{A}$ be an L_1-automaton. Then $\mathbb{A}$ is equivalent to $\mathbb{A}^\sharp$.*

A key proposition, relating the various formulas, languages and substitutions that feature in the simulation construction, is the following.

Proposition 7.57 *Let $\mathbb{A}$ be an L_1-automaton and let D be some set. Suppose that for each $a \in A$ a marking $m_a : D \to \wp A$ is given. For $R \in A^\sharp$, let $m_R : D \to \wp(A \times A)$ and $m^\sharp_R : D \to \wp(A^\sharp)$ be the markings given by*

$$\begin{array}{rcl} m_R(d) & := & \{(a,b) \mid a \in \mathsf{Ran}(R) \ \&\ b \in m_a(d)\} \\ m^\sharp_R(d) & := & \{m_R(d)\}. \end{array}$$

Then the following are equivalent, for any $c \in \wp\mathsf{P}$:

1. *$(D, m_a) \Vdash^1 \Theta(a, c)$ for each $a \in \mathsf{Ran}(R)$;*
2. *$(D, m_R) \Vdash^1 \bigwedge\{\Theta^\star(a, c) \mid a \in \mathsf{Ran}(R)\}$;*
3. *$(D, m^\sharp_R) \Vdash^1 \Theta^\sharp(R, c)$.*

We leave the (straightforward) proof of this Proposition as an exercise to the reader.

Proof of Proposition 7.56. Fix an arbitrary pointed model $(\mathbb{S}, s_0)$, then it suffices to prove that

$$\mathbb{A} \text{ accepts } (\mathbb{S}, s_0) \text{ iff } \mathbb{A}^\sharp \text{ accepts } (\mathbb{S}, s_0). \tag{56}$$

For the direction from left to right, define a position (R, s) to be *safe* if for all $a \in \mathsf{Ran}(R)$, (a, s) is winning for $\exists$ in the acceptance game $\mathcal{G} = \mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, s_0)$. Now define the following strategy for $\exists$ in $\mathcal{G}^\sharp = \mathcal{A}(\mathbb{A}^\sharp, \mathbb{S})@(R_I, s_0)$:

- If (R, s) is safe, then $\exists$ uses Proposition 7.57 to transform the set of moves $\{m_{a,s} \mid a \in \mathsf{Ran}(R)\}$, given by her winning strategy in $\mathcal{G}$, into a marking $m^\sharp_{R,s} : R[s] \to \wp A^\sharp$.
- If (R, s) is not safe, then $\exists$ plays in a random way.

It is not very hard to prove the following three claims on this strategy.

Claim 1 If (R, s) is safe then the moves suggested by the above strategy are legitimate.

Claim 2 If (R, s) is safe then all pairs (Q, t) such that $Q \in m^\sharp_{R,s}(t)$ are safe.

Claim 3 Consider an infinite $\mathcal{G}^\sharp$-match, guided by the above strategy for $\exists$, with basic positions $(R_I, s_0)(R_1, s_1)(R_2, s_2)\ldots$, and let $a_I a_I a_1 a_2 \ldots$ be a trace through $R_I R_1 R_2 \ldots$ Then there is an infinite $\mathcal{G}$-match, guided by $\exists$'s winning strategy, of which the basic positions are $(a_I, s_0)(a_1, s_1)(a_2, s_2)\ldots$

On the basis of these three claims, it easily follows that the given strategy is winning for $\exists$ from any safe position. In particular, it follows from the assumption that $(a_I, s_0) \in \mathrm{Win}_\exists(\mathcal{G})$ that (R_I, s_0) is safe, and hence winning for $\exists$ in $\mathcal{G}^\sharp$. This shows that $\mathbb{A}^\sharp$ accepts $(\mathbb{S}, s_0)$, as required.

The proof of the opposite direction ('$\Leftarrow$') of (56) is somewhat similar, and left as an exercise. QED

Regular automata

In the previous subsection we defined a nondeterministic automaton $\mathbb{A}^\sharp$ and proved it to be equivalent to the given automaton $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$. The only shortcoming of the automaton $\mathbb{A}^\sharp$ is that its acceptance condition $\mathrm{NBT}_\Omega \subseteq (A^\sharp)^\omega$ is not given by a parity function. We will now see that this problem can easily be overcome since NBT_Ω has the form of an ω-regular language over the alphabet $A^\sharp$, that is, it is recognized by some stream automaton.

Definition 7.58 An automaton $\mathbb{A} = \langle A, \Theta, Acc, a_I \rangle$ is called *ω-regular* if $Acc \subseteq A^\omega$ is an ω-regular language, i.e., if Acc is the stream language recognized by some deterministic stream automaton with a parity (or Muller) acceptance condition. $\lhd$

Here we shall prove that, given an regular automaton $\mathbb{A}$ of which the acceptance condition is given by some deterministic parity stream automaton $\mathbb{Z}$, we can effectively construct a parity automaton $\mathbb{A} \odot \mathbb{Z}$ that is equivalent to $\mathbb{A}$. First, however, we show that, indeed, $\mathbb{A}^\sharp$ is a regular automaton, by constructing a stream automaton recognizing the ω-language NBT_Ω.

Proposition 7.59 *Let A be some finite set, and let $\Omega : A \to \omega$ be a parity function on A. Then the set NBT_Ω is an ω-regular language over the alphabet $A^\sharp$.*

Proof. First we define a nondeterministic $A^\sharp$-stream parity automaton $\mathbb{B}$ which accepts exactly those infinite $A^\sharp$-streams that *do* contain a bad trace. Given the properties of parity stream automata it is fairly straightforward to continue from here. First, take a deterministic equivalent $\mathbb{B}'$ of $\mathbb{B}$; such an automaton exists by Theorem 4.27. And second, since $\mathbb{B}'$ is deterministic, it is easy to perform complementation on it, that is, define an automaton $\mathbb{C}$ that accepts exactly those $A^\sharp$-streams that are rejected by $\mathbb{B}'$. In short: $L_\omega(\mathbb{C}) = (A^\sharp)^\omega \setminus L_\omega(\mathbb{B}') = (A^\sharp)^\omega \setminus L_\omega(\mathbb{B})$. Clearly then $L_\omega(\mathbb{C}) = \mathrm{NBT}_\Omega$.

For the definition of $\mathbb{B}$, take an object $b_I \notin A$, and define $B := A \cup \{b_I\}$. Let $\Delta : B \times A^\sharp \to \wp(B)$ be given by putting

$$\Delta(b, R) := \begin{cases} \mathsf{Ran}(R) & \text{if } b = b_I, \\ R[b] & \text{if } b \in A, \end{cases}$$

and define Ω^{+1} by putting $\Omega^{+1}(a) := \Omega(a) + 1$ for $a \in A$, and $\Omega^{+1}(b_I) := 0$. Then $\mathbb{B}$ is the automaton $\langle B, \Delta, \Omega^{+1}, b_I \rangle$.

It is immediate from the definitions that $b_I \xrightarrow{R} a$ iff $a \in \mathsf{Ran}(R)$, that is, if there is some $a' \in A$ such that $a'Ra$. From this and the definition of Δ it follows that

$$b_I \xrightarrow{R_1} a_1 \xrightarrow{R_2} a_2 \xrightarrow{R_3} \dots$$

is a run of $\mathbb{B}$ iff there is some $a_0 \in A$ such that $a_0 a_1 a_2 \dots$ is a trace through $R_1 R_2 \dots$ Then the definition of Ω^{+1} ensures that $\mathbb{B}$ indeed accepts those $A^\sharp$-streams that contain a bad trace. QED

It follows from Proposition 7.59 that the automaton $\mathbb{A}^\sharp$ defined in the previous section is a regular automaton. Hence we have proved the main result of this section if we can show that every disjunctive regular automaton can be replaced by a disjunctive modal automaton with a parity acceptance condition. This is what we will focus on now. In fact, we will effectively transform a nondeterministic, regular automaton $\mathbb{A}$ (of which the acceptance condition is given as the stream language recognized by some stream automaton $\mathbb{Z}$) into an equivalent parity automaton $\mathbb{A} \odot \mathbb{Z}$.

Definition 7.60 Let $\mathbb{Z} = \langle Z, \zeta, \Omega, a_I \rangle$ be a deterministic parity A-stream automaton, and let $\mathbb{A} = \langle A, \Theta, Acc, a_I \rangle$ be a disjunctive modal automaton. Then $\mathbb{A} \odot \mathbb{Z}$ is the disjunctive modal automaton given as

$$\mathbb{A} \odot \mathbb{Z} = \langle A \times Z, \Theta^{\zeta}, \Psi, (a_I, z_I) \rangle,$$

where $\Theta^{\zeta} : \big((A \times Z) \times \wp \mathsf{P}\big) \to \mathtt{1DML}(A \times Z)$ is given by

$$\Theta^{\zeta}\big((a, z), c\big) := \Theta(a, c)[(b, \zeta(z, a))/b \mid b \in A],$$

and

$$\Psi\big(a, z) := \Omega(z).$$

defines $\Psi : A \times Z \to \omega$. $\lhd$

Intuitively, the automaton $\mathbb{A} \odot \mathbb{Z}$ behaves like $\mathbb{A}$, with the stream automaton $\mathbb{Z}$ following and directly processing the path through $\mathbb{A}$ taken during a match of the acceptance game. More precisely, when the automaton $\mathbb{A}$ moves from state a to b, the corresponding moves of $\mathbb{A} \odot \mathbb{Z}$ are from any position (a, z) to $(b, \zeta(z, a))$, where $\zeta(z, a)$ is the state obtained from z by processing the 'letter' a. Formally, this is established by the transition structure Θ^{ζ} of the automaton $\mathbb{A} \odot \mathbb{Z}$ as follows: $\Theta^{\zeta}\big((a, z), c\big)$ is obtained from $\Theta(a, c)$ by substituting every occurrence of a $b \in A$ by the ('formal') variable $(b, \zeta(z, a)) \in A \times Z$.

Theorem 7.61 *Let $\mathbb{Z} = \langle Z, \zeta, \Omega, z_I \rangle$ be a deterministic parity stream automaton, and let $\mathbb{A} = \langle A, \Theta, Acc, a_I \rangle$ be a disjunctive modal automaton such that $Acc = L_{\omega}(\mathbb{Z})$. Then $\mathbb{A}$ and $\mathbb{A} \odot \mathbb{Z}$ are equivalent.*

▶ `Proof of Theorem 7.61 to be supplied`

Finally, for the proof of the Simulation Theorem we need to combine various results obtained in this Chapter.

Proof of Theorem 7.50. It follows from the Propositions 7.49, 7.56 and 7.59 that every modal automaton can be simulated by a disjunctive, regular automaton. Then the Simulation Theorem follows by combining this observation with Theorem 7.61. QED

Notes

▶ `TBS`

Exercises

Exercise 7.1 Show that the 'slow' acceptance discussed in Remark 7.10 is equivalent to the standard acceptance game of Definition 7.5.

Exercise 7.2 Give a direct, game-theoretic argument proving Theorem 7.12. That is, show that modal automata are bisimulation invariant.

Exercise 7.3 Show the equivalence of the two notions of disjunctive modal automata as discussed in Remark 7.14. That is, give a construction that transforms an arbitrary disjunctive modal automaton into a $\mathtt{1DML}_r$-automaton.

Exercise 7.4 Let $\mathbb{A}$ be a disjunctive modal automaton, and let $(\mathbb{S}, r)$ be a finite pointed Kripke model. Show that $\mathbb{S}, r \Vdash \mathbb{A}$ iff there is a *finite* pointed model $(\mathbb{S}', r')$ such that $\mathbb{S}, r \leftrightarroweq (\mathbb{S}', r')$ and $\mathbb{S}', r' \Vdash_s \mathbb{A}$.

Exercise 7.5 Show that the one-step languages $\mathtt{1FO}$ and $\mathtt{1FOE}$ are closed under taking boolean duals.

Exercise 7.6 Prove Proposition 7.38

Exercise 7.7 Prove Proposition 7.57.

Exercise 7.8 Prove equivalence (56) in the proof of Proposition 7.56.

8 Model theory of the modal μ-calculus

In this Chapter we will see how to apply the automata-theoretic tools developed in the previous chapter to prove some model-theoretic results about the modal μ-calculus.

▶ `overview of chapter to be supplied`

8.1 Small model property

As our first result we will prove a small model property for the modal μ-calculus, by showing that if a modal automaton accepts some pointed Kripke model, it accepts one of which the size is bounded by the size of the automaton. Recall that, given a modal automaton $\mathbb{A}$ we refer to the class of pointed Kripke models that are accepted by $\mathbb{A}$ as the *query* of $\mathbb{A}$, notation: $\mathcal{Q}(\mathbb{A})$, and that classes of this form are called *recognizable*.

Theorem 8.1 *Let $\mathbb{A}$ be a modal automaton. Then $\mathcal{Q}(\mathbb{A}) \neq \varnothing$ iff $\mathbb{A}$ accepts a finite pointed model of size at most exponential in the state-size of $\mathbb{A}$.*

Because of the Simulation Theorem it suffices to prove Theorem 8.1 for *disjunctive* modal automata. Our proof will be based on an alternative perspective of these devices, revealing their close resemblance the Kripke models that they operate on.

Kripke automata

The key observation in our proof is that the semantics of the cover modality and the notion of a bisimulation are defined in a very similar fashion, both involving the coalgebraic presentation of Kripke models, and the notion of *relation lifting*.

Fix a set P of proposition letters. Recall from Remark 1.3 and Definition 1.4 that we can represent a Kripke model[4] (S, R, V) as a pair

$$\mathbb{S} = (S, \sigma : S \to \mathsf{K}S),$$

where K is the Kripke functor given by putting, for an arbitrary set S:

$$\mathsf{K}S := \wp(\mathsf{P}) \times \wp(S).$$

In Definition 1.28 we introduced two notions of *relation lifting*. Given a binary relation $Z \subseteq S \times S'$, we define the relation $\overline{\wp}Z \subseteq \wp S \times \wp S'$ as follows:

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

Similarly, define, associated with the Kripke functor K, the relation $\overline{\mathsf{K}}Z \subseteq \mathsf{K}S \times \mathsf{K}S'$ as follows:

$$\overline{\mathsf{K}}Z := \{((\pi, X), (\pi', X')) \mid \pi = \pi' \text{ and } (X, X') \in \overline{\wp}Z\}.$$

[4] We restrict to the monomodal case in this section.

Position	Player	Admissible moves
$(a, s) \in A \times S$	-	$\{(\alpha(a), \sigma(s))\}$
$(\beta, \tau) \in \mathsf{K}A \times \mathsf{K}S$	$\exists$	$\{Z \in \wp(A \times S) \mid (\beta, \tau) \in \overline{\mathsf{K}}Z\}$
$Z \in \wp(A \times S)$	$\forall$	$Z = \{(b, t) \mid (b, t) \in Z\}$

Table 13: Bisimilarity game for Kripke models

To make our point we now introduce a new class of automata, consisting of so-called *Kripke automata*, and show that these are in fact equivalent to the disjunctive automata defined earlier on.

As our starting point we consider, for two Kripke models $\mathbb{A} = \langle A, \alpha\rangle$ and $\mathbb{S} = \langle S, \sigma\rangle$, the bisimilarity game $\mathcal{B}(\mathbb{A}, \mathbb{S})$ of Definition 1.25. Using the above notion of relation lifting, the rules of this game can be reformulated as in Table 13. Recall that the winning conditions of the bisimilarity game are such that all infinite games are won by $\exists$.

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 carrier set A) to be finite.

(2) Second, and equally undramatic, we 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* to the transition structure of $\mathbb{A}$. That is, Kripke automata will harbour many 'realizations' of Kripke models — and in each round of the acceptance game, it is $\exists$'s task to pick an actual local realization of the current state of $\mathbb{A}$.

Definition 8.2 Given a set P of proposition letters, a *Kripke automaton* for P is a quadruple $\mathbb{A} = \langle A, \Delta, \Omega, a_I\rangle$ such that the transition function Δ is given as a map $\Delta : A \to \wp(\mathsf{K}A)$. 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 14. A pointed Kripke model $(\mathbb{S}, s)$ is *accepted* by $\mathbb{A}$ if the position

Position	Player	Admissible moves	Priority
$(a, s) \in A \times S$	$\exists$	$\{(\gamma, \sigma(s)) \in \mathsf{K}A \times \mathsf{K}S \mid \gamma \in \Delta(a)\}$	$\Omega(a)$
$(\gamma, \tau) \in \mathsf{K}A \times \mathsf{K}S$	$\exists$	$\{Z \subseteq A \times S \mid (\gamma, \tau) \in \overline{\mathsf{K}}Z\}$	0
$Z \in \wp(A \times S)$	$\forall$	Z	0

Table 14: Acceptance game for Kripke automata

(a_I, s) is a winning position for $\exists$ in the acceptance game. $\lhd$

For an informal description of the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$, note that each round consists of exactly three moves, with interaction pattern $\exists\exists\forall$. At a basic position (a, s), the 'K-unfolding' $\sigma(s) \in \mathsf{K}S$ of s is fixed, but $\exists$ *chooses* the unfolding of a to be an arbitrary element

γ of $\Delta(a)$. After this move, the play arrives at a position of the form $(\gamma, s) \in \mathsf{K}A \times S$. The players now proceed as in the bisimilarity game for Kripke models. First $\exists$ chooses a 'local bisimulation' linking γ and $\sigma(s)$, that is, a relation $Z \subseteq A \times S$ such that $(\gamma, \sigma(s)) \in \overline{\mathsf{K}}Z$. Spelled out, this means that $\exists$ can only choose such a relation Z if γ is of the form $(c, B) \in \wp(\mathsf{P}) \times \wp(A)$ with $c = \sigma_V(s)$, and that Z has to satisfy the back and forth conditions, stating that for all $b \in B$ there is $t \in R[s]$ with bZt, and vice versa. The round ends with $\forall$ choosing an element (b, t) from Z, thus providing the next basic position of the match.

We will now show that Kripke automata are nothing but disjunctive automata in disguise, and vice versa.

Definition 8.3 First let $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ be some Kripke automaton. We define its modal companion $\mathbb{A}^M$ as the disjunctive modal automaton $\mathbb{A}^M := \langle A, \Delta^M, \Omega, a_I \rangle$, where $\Delta^M : A \times \wp(\mathsf{P}) \to \mathtt{1DML}(A)$ is given by putting

$$\Delta^M(a, c) := \bigvee \{\nabla B \mid (c, B) \in \Delta(a)\}.$$

Conversely, let $\mathbb{D} = \langle D, \Theta, \Omega, d_I \rangle$ be a disjunctive modal automaton. Without loss of generality we may assume that the domain of Θ consists of formulas in the restricted format of Remark 7.14, that is, for every pair $(a, c) \in A \times \wp(\mathsf{P})$ there is a (possibly empty) index set $I_{a,c}$ such that

$$\Theta(a, c) = \bigvee \{\nabla B_i \mid i \in I_{a,c}\}.$$

We now define the transition map Δ_Θ by putting

$$\Delta_\Theta(a) := \{(c, B_i) \in \mathsf{K}A \mid c \in \wp(\mathsf{P}), i \in I_{a,c}\},$$

and define $\mathbb{D}^K := \langle D, \Delta_\Theta, \Omega, d_I \rangle$ and call this structure the *Kripke companion* of $\mathbb{D}$. $\lhd$

Remark 8.4 For a better understanding of the equivalence between disjunctive modal automata and Kripke models, it may be useful to take the following perspective. Given sets P (of proposition letters) and A of states, it is not hard to see that the collection of possible transition functions of disjunctive modal automata (in the restricted format of Remark 7.14 corresponds to the set

$$T_D := \big(A \times \wp(\mathsf{P})\big) \to \wp(\wp(A)),$$

while the set of possible transition maps of Kripke automata is given as the collection

$$T_K := A \to \wp\big(\wp(\mathsf{P}) \times \wp(A)\big).$$

Now recall that by 'currying' there is a bijective correspondence
(†) $(X \times Y) \to Z \;\cong\; X \to (Y \to Z)$
for any triple of sets X, Y and Z. Furthermore, for any set X there is a well-known bijective correspondence between the powerset $\wp(X)$ of X and the collection of functions from X to the two-element set $2 := \{0, 1\}$:
(‡) $\wp(X) \;\cong\; X \to 2$.

Using these observations it is straightforward to verify the following bijective correspondences between the sets T_D and T_K:

$$\begin{array}{ll}
 & \big(A \times \wp(\mathsf{P})\big) \to \wp\wp(A) \\
\cong (\ddagger) & \big(A \times \wp(\mathsf{P})\big) \to \big(\wp(A) \to 2)\big) \\
\cong (\dagger) & \big(A \times \wp(\mathsf{P}) \times \wp(A)\big) \to 2 \\
\cong (\dagger) & A \to \Big(\big(\wp(\mathsf{P}) \times \wp(A)\big) \to 2\Big) \\
\cong (\ddagger) & A \to \wp\big(\wp(\mathsf{P}) \times \wp(A)\big)
\end{array}$$

In fact, the translations given in Definition 8.3 can be obtained by computing the bijections between T_D and T_K, on the basis of those in (†) and (‡). $\lhd$

Proposition 8.5 *(i) Let $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ be a Kripke automaton. Then $\mathbb{A} \equiv \mathbb{A}^M$.*
(ii) Let $\mathbb{D} = \langle D, \Theta, \Omega, d_I \rangle$ be a disjunctive modal automaton. Then $\mathbb{D} \equiv \mathbb{D}^K$.

Proof. The proof of this proposition is straightforward. If we merge the two moves of $\exists$ in each round of the acceptance game for Kripke automata into one, we may in fact show that, for any Kripke model $\mathbb{S}$, the acceptance games $\mathcal{A}(\mathbb{A}^M, \mathbb{S})$ and $\mathcal{A}(\mathbb{A}, \mathbb{S})$ are *isomorphic*, and similarly for the acceptance games $\mathcal{A}(\mathbb{D}^K, \mathbb{S})$ and $\mathcal{A}(\mathbb{D}, \mathbb{S})$. QED

Small model property for Kripke automata

We will now prove the small model property for *Kripke automata*. This framework allows us to prove a result that is quite a bit stronger than just a small model theorem: we may show that, if $\mathbb{A}$ is a Kripke automaton recognizing a non-empty query, then $\mathcal{Q}\mathbb{A}$ contains a Kripke model that 'lives inside' or *inhabits* $\mathbb{A}$.

Definition 8.6 Let $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$ be a Kripke automaton. If S is a subset of A, and $\sigma : S \to \mathsf{K}S$ is such that $\sigma(s) \in \Delta(s)$ for all $s \in S$, then we say that the Kripke model $\mathbb{S} = \langle S, \sigma \rangle$ *inhabits* $\mathbb{A}$. When we use this terminology for a pointed Kripke model $(\mathbb{S}, s)$, we require in addition that $s = a_I$. $\lhd$

The key tool in our proof of the small model property will be the following *satisfiability game* that we may associate with a Kripke automaton. Intuitively the reader may think of this game as the simultaneous projection on $\mathbb{A}$ of all acceptance games of $\mathbb{A}$, as should become clear from the proof of Theorem 8.8 below.

Definition 8.7 Let $\mathbb{A} = \langle A, \Delta, \Omega, a_I \rangle$ be a Kripke automaton. Then the *satisfiability game* $\mathcal{S}(\mathbb{A})$ is given by Table 15. The winning condition for infinite matches is defined using the priority map for game positions (see the table) as a parity condition. $\lhd$

One last remark before we formulate and prove the main technical result of this section: the proof of this theorem involves a crucial application of the Positional Determinacy of parity games.

Position	Player	Admissible moves	Priority
$a \in A$	$\exists$	$\Delta(a)$	$\Omega(a)$
$(c, B) \in \mathsf{K}A$	$\forall$	B	0

Table 15: Satisfiability game for Kripke automata

Theorem 8.8 *The following are equivalent, for any Kripke automaton $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$:*
1) $\mathcal{Q}(\mathbb{A}) \neq \varnothing$;
2) $a_I \in \mathrm{Win}_{\exists}(\mathcal{S}(\mathbb{A}))$;
3) $\mathbb{A}$ accepts a pointed model inhabiting $\mathbb{A}$.

Proof. $\boxed{1 \Rightarrow 2}$ Suppose that $\mathbb{A}$ accepts some pointed model $(\mathbb{S}, s_0)$. Then by definition, $\exists$ has a winning strategy in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, s_0)$. This strategy will be the basis of her winning strategy in the satisfiability game of $\mathbb{A}$.

Concretely, in $\mathcal{S}(\mathbb{A})@a_I$, $\exists$ will maintain the following condition. Put $a_0 = a_I$, and let

$$a_0(c_1, B_1)a_1(c_2, B_2)\ldots a_k,$$

be an initial segment of an $\mathcal{S}(\mathbb{A})$-match (with $(c_{i+1}, B_{i+1}) \in \Theta(a_i)$ being the move of $\exists$ at position a_i, and $a_{i+1} \in B_{i+1}$ the next move of $\forall$). Then $\exists$ sees this match as the projection of a parallel match of $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, s_0)$ where she plays her winning strategy:

$$\begin{array}{cccccccccc}
(a_0, s_0) & ((c_1, B_1), s_0) & Z_1 & (a_1, s_1) & \ldots & (a_k, s_k) & ((c_{k+1}, B_{k+1}), s_k) & Z_{k+1} & \ldots \\
\Downarrow & \Downarrow & \Downarrow & \Downarrow & & \Downarrow & \Downarrow & \Downarrow & \\
a_0 & (c_1, B_1) & - & a_1 & \ldots & a_k & (c_{k+1}, B_{k+1}) & - & \ldots
\end{array}$$

The existence of such a parallel match is easily proved by an inductive argument, of which the base case is immediate by the shape (a_I versus (a_I, s_0)) of the initial game positions. Inductively assume that at stage k, the matches of $\mathcal{S}(\mathbb{A})$ and $\mathcal{A}(\mathbb{A}, \mathbb{S})$ have arrived at the positions a_k and (a_k, s_k) respectively. We will show that there is a way to continue both matches for one round in such a way that the next basic positions are of the form b and (b, t), respectively, for some $b \in A$ and $t \in S$, with the continuation in the acceptance game being guided by $\exists$'s winning strategy.

Suppose that $\exists$'s winning strategy in the acceptance game tells her to choose position $((c, B), \sigma(s_k))$, followed by the relation Z. Then at position a_k of $\mathcal{S}(\mathbb{A})$, we define her strategy to be such that she picks (c, B). Now suppose that in the match of $\mathcal{S}(\mathbb{A})$, $\forall$ chooses some element $b \in B$ as the next position. It follows by the assumption that $\exists$'s strategy is winning, that $(c, B) \in \Theta(a_k)$, $c = \sigma_V(s_k)$ and $(B, R[s_k]) \in \overline{\wp}(Z)$. Hence there must be an element $t \in R[s_k]$ such that $(b, t) \in Z$; in the acceptance game, she may look at a continuation of the match where $\forall$ picks the pair (b, t). In other words, we have proved that $\exists$ can maintain the parallel match for one more round.

Using this strategy in the satisfiability game will then guarantee her to win the match, since the associated sequence of $\mathbb{A}$-states is the same for both matches, and in the $\mathcal{A}(\mathbb{A}, \mathbb{S})$-match $\exists$ plays according to a strategy that was assumed to be winning.

$\boxed{2 \Rightarrow 3}$ Assume that $\exists$ has a winning strategy in the satisfiability game starting from the initial state a_I of $\mathbb{A}$. Let $S := \mathrm{Win}_\exists(\mathcal{S}(\mathbb{A}))$ be the set of positions in A that are winning for $\exists$. The key point of the satisfiability game for Kripke automata is that $\mathcal{S}(\mathbb{A})$ is a parity game, and so we may without loss of generality assume that this strategy is *positional*, see Theorem 5.22. In other words, we may represent it as a map $\sigma : S \to \mathsf{K}A$. We invite the reader to check that $\sigma(a) \in \mathsf{K}S$ for all $a \in S$. Now define $\mathbb{S}$ be the Kripke model $\langle S, \sigma \rangle$. The map $\sigma : S \to \mathsf{K}S$ then induces a binary relation $R \subseteq S \times S$ and a valuation $V : \mathsf{P} \to \wp(S)$, viz., the unique R and V such that $\sigma(s) = (R[s], \sigma_V(s))$. We claim that $\mathbb{A}$ accepts $(\mathbb{S}, a_I)$.

To see why this is the case, we will prove that (a_I, a_I) is a winning position in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$. The winning strategy that we may equip $\exists$ with in this game is in fact very simple:

- at position (a, s), pick $(\sigma(a), \sigma(s))$ as the next position if $a = s \in \mathrm{Win}_\exists(\mathcal{S}(\mathbb{A}))$, and choose a random element otherwise;
- at position $((c, B), (c', B'))$, pick the relation $\{(b, b) \mid b \in B \cap B'\}$.

It can be proved that any match of the acceptance game in which $\exists$ uses this strategy, can be 'projected' onto a match of the satisfiability game in which she plays her winning strategy:

$$\begin{array}{cccccccc}
(a_I, a_I) & (\sigma(a_I), \sigma(a_I)) & \{(b, b) \mid b \in R[a_I]\} & (a_1, a_1) & (\sigma(a_1), \sigma(a_1)) & \ldots & (a_n, a_n) & \ldots \\
\Downarrow & \Downarrow & \Downarrow & \Downarrow & \Downarrow & & \Downarrow & \\
a_I & \sigma(a_I) & - & a_1 & \sigma(a_1) & \ldots & a_n & \ldots
\end{array}$$

Given the winning conditions of $\mathcal{A}(\mathbb{A}, \mathbb{S})$ and $\mathcal{S}(\mathbb{A})$ it is then immediate that the given strategy indeed guarantees that $\exists$ wins any match starting at position (a_I, a_I).

$\boxed{3 \Rightarrow 1}$ This implication is a direct consequence of the definitions. QED

8.2 Normal forms and decidability

In this section we will see two more corollaries of the results in the previous chapter.

Disjunctive normal form

As a first consequence, we now see that every formula of the modal μ-calculus can be brought into so-called *disjunctive normal form*. For the definition of the connectives used below we refer to Definition 1.35.

Definition 8.9 Given sets P of proposition letters, the set of *disjunctive* modal μ-calculus formulas over P is given by the following grammar:

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

Here $\pi \in \mathtt{CL}(\mathsf{P})$ denotes a conjunction of literals over P, and Φ a finite collection of disjunctive formulas, and x is a variable *not* in P.

We let $\mu\mathtt{ML}_\mathsf{D}(\mathsf{P})$ denote the sentences of this language, that is, the disjunctive formulas φ such that $FV(\varphi) \subseteq \mathsf{P}$. $\lhd$

These formula are called disjunctive because the only admissible conjunctions are the special ones of the form $\pi \bullet \nabla\Phi$, where π is a propositional formula (in fact, a conjunction of literals).

Theorem 8.10 *There is an effective algorithm that rewrites a modal fixpoint formula $\xi \in \mu\mathtt{ML}(\mathsf{P})$ into an equivalent disjunctive formula ξ^d of closure size at most exponential in $|\xi|$.*

- ▶ `proof (based on the results of the previous chapters) to be supplied.`
- ▶ `size issues to be addressed!`

Decidability

- ▶ `Intro`

Theorem 8.11 *There is an algorithm that decides in linear time (measured in dag-size) 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 are satisfiable;
4. $\pi \bullet \nabla\Phi$ is satisfiable iff both π and each $\varphi \in \Phi$ is satisfiable;
5. if $\mu x.\varphi$ is disjunctive, then it is satisfiable iff $\varphi[\bot/x]$ is satisfiable;
6. if $\nu x.\varphi$ is disjunctive, then it 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.

Corollary 8.12 *There is an algorithm that decides in elementary time whether a given modal fixpoint formula ξ is satisfiable or not.*

- ▶ `Corollary 8.12 does not provide the best complexity bound for the satisfiability problem for the` μ`-calculus, which can in fact be solved in (singly) exponential time.`

8.3 Uniform interpolation and bisimulation quantifiers

In this section we will prove that the modal μ-calculus enjoys the property of *uniform interpolation* by proving that we can express the so-called *bisimulation quantifiers* in the language.

Definition 8.13 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 $(\mathbb{S}, s)$. $\lhd$

A formalism has the *(Craig) 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 proposition letters that occur both in φ and ψ, or more precisely: $FV(\theta) \subseteq FV(\varphi) \cap FV(\psi)$.

► `why this is an important property`

Uniform interpolation is a very strong version of interpolation in which the interpolant θ does not depend on the particular shape of one of the formulas, but only on its *vocabulary* (set of free variables). More precisely, we define the following.

Definition 8.14 Let φ be a modal fixpoint formula, and $\mathsf{P} \subseteq FV(\varphi)$ be a set of variables. Then a *(right) uniform interpolant* of φ with respect to P is a formula θ with $FV(\theta) \subseteq \mathsf{P}$, such that

$$\varphi \models \psi \text{ iff } \theta \models \psi. \tag{57}$$

for all formulas ψ with $FV(\psi) \cap FV(\varphi) \subseteq \mathsf{P}$. $\lhd$

In words, (57) states that θ has exactly the same consequences as φ, at least, if we restrict to formulas ψ such that all free variables shared by φ and ψ belong to P.

Remark 8.15 To justify the terminology 'uniform interpolant', take some formula ψ with $FV(\psi) \cap FV(\varphi) \subseteq \mathsf{P}$. We claim that

$$\varphi \models \psi \text{ implies } \varphi \models \theta \text{ and } \theta \models \psi \tag{58}$$

for any uniform interpolant θ of φ with respect to P.

To see this, suppose that $\varphi \models \psi$, and let θ be a uniform interpolant of φ with respect to P. Then we have $\theta \models \psi$ by (57), so it remains to show that $\varphi \models \theta$. But this follows immediately from the fact that by definition we have $FV(\theta) \cap FV(\varphi) \subseteq \mathsf{P}$, so that we may apply (57) to θ itself (and use that, obviously, $\theta \models \theta$). $\lhd$

Remark 8.16 Dually, we could have introduced the notion of a *left* uniform interpolant for ψ, instead of a *right* interpolant for φ. A left interpolant for ψ, with respect to a set $\mathsf{P} \subseteq FV(\psi)$ of proposition letters, is a formula χ with $FV(\chi) \subseteq \mathsf{P}$, and such that $\varphi \models \psi$ iff $\varphi \models \chi$. But since negation is definable in the modal μ-calculus as an operation $\sim : \mu\mathtt{ML}(\mathsf{P}) \to \mu\mathtt{ML}(\mathsf{P})$ and so we have $\varphi \models \psi$ iff $\sim\psi \models \sim\varphi$, it is not hard to see that if θ is a (right) uniform interpolant for ψ, then its negation $\sim\theta$ is a left interpolant for ψ. In other words, since our language is closed under classical negation, requiring that every formula has a right uniform interpolant is equivalent to requiring that every formula has a left uniform interpolant. $\lhd$

The following theorem states that uniform interpolants exist in the modal μ-calculus.

Theorem 8.17 (Uniform Interpolation) *Let φ be a modal fixpoint formula, and let P be a set of variables such that $\mathsf{P} \subseteq FV(\varphi)$. Then φ has a uniform interpolant with repect to P.*

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

Definition 8.18 Given a proposition letter q, the *bisimulation quantifier* $\widetilde{\exists} q$ is an operator with the following semantics:

$$\mathbb{S}, s \Vdash \widetilde{\exists} q.\varphi \text{ iff } \mathbb{S}', s' \Vdash \varphi, \text{ for some pointed model } \mathbb{S}', s' \leftrightarroweq_{\mathsf{R}\setminus q} \mathbb{S}, s, \tag{59}$$

where $\mathbb{S}$ is some Kripke model over a set R of proposition letters, and $\leftrightarroweq_{\mathsf{R}\setminus q}$ is the bisimilarity relation 'up to q', that is, we only require the condition (prop) of Definition 1.18 to hold for proposition letters $p \in \mathsf{R} \setminus q$. $\lhd$

The bisimulation quantifier $\widetilde{\exists} q$ is a second-order existential quantifier, but nonstandard in the sense that it does not quantify over subsets of the actual model $\mathbb{S}$, but rather over subsets of possibly distinct (but bisimilar-up-to-q) models. For instance, if s is a state in $\mathbb{S}$ with one single successor, then obviously the formula $\widetilde{\exists} q(\Diamond q \wedge \Diamond \overline{q})$ would be false if we had to interpret q as a subset of S. However, taking a bisimilar pointed model $(\mathbb{S}', s')$ such that s' has two successors, we can easily interpret q as a subset of S' such that the formula $\Diamond q \wedge \Diamond \overline{q}$ becomes true at s'. Similarly, the formula $\widetilde{\exists} q(q \wedge \Box \overline{q})$ holds at any point in any Kripke model.

The main result underlying the proof of Theorem 8.17 is that the bisimulation quantifiers are *definable* in the modal μ-calculus. The following notation will be convenient.

Convention 8.19 Where P is a set of proposition letters, and q is a proposition letter (which may or may not belong to P), we write $\mathsf{P} \setminus q$ rather than $\mathsf{P} \setminus \{q\}$.

Theorem 8.20 *For any set P of proposition letters, and any proposition letter q, there is a map*

$$\widetilde{\exists} q : \mu\mathtt{ML}_{\mathsf{D}}(\mathsf{P}) \to \mu\mathtt{ML}_{\mathsf{D}}(\mathsf{P} \setminus q)$$

such that for any formula $\varphi \in \mu\mathtt{ML}_{\mathsf{D}}(\mathsf{P})$, we have $FV(\widetilde{\exists} q.\varphi) = FV(\varphi) \setminus q$, and the semantics of $\widetilde{\exists} q.\varphi$ satisfies (59)*, for any Kripke model over a set of proposition letters $\mathsf{R} \supseteq \mathsf{P}$.*

The proof of Theorem 8.20 crucially involves *disjunctive* modal automata. Before going into the details, there is a technicality that we need to get out of the way.

Remark 8.21 Let $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$ be a modal automaton over some set P of proposition letters, and let $\mathbb{S} = (S, R, V)$ be a Kripke model over some, possibly larger, set R. Then strictly speaking the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$ is not well-defined since the domain of the transition map Θ is of the form $\mathsf{Dom}(\Theta) = A \times \wp(\mathsf{P})$, while the range of the colouring map σ_V of $\mathbb{S}$ is the set $\mathsf{Ran}(\sigma_V) = \wp(\mathsf{R})$. But clearly we can take care of this mismatch by working with the map $\Theta_{\mathsf{R}} : A \times \wp(\mathsf{R}) \to \mathtt{1ML}(A)$ given by

$$\Theta_{\mathsf{R}}(a, c) := \Theta(a, c \cap \mathsf{P}).$$

In the sequel we will largely ignore this issue. $\lhd$

We now turn to the details of the proof of Theorem 8.20. Because of the existence of truth-preserving translations between formulas and automata, it suffices to provide a construction on modal automata that instantiates the bisimulation quantifier, and because of the Simulation Theorem it suffices to define this construction for disjunctive modal automata.

Definition 8.22 Let P be a set of proposition letters and let q be a proposition letter (possibly but not necessarily in P). Let $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$ be a disjunctive modal automaton over the set P. We abbreviate $C := \wp(\mathsf{P})$ and $C^- := \wp(\mathsf{P} \setminus \{q\})$.

Now we define the modal automaton $\widetilde{\exists} q.\mathbb{A}$ as the structure $\widetilde{\exists} q.\mathbb{A} := \langle A, \Theta^{\pm q}, \Omega, a_I \rangle$, where

$$\Theta^{\pm q}(a, c) := \Theta(a, c \setminus \{q\}) \vee \Theta(a, c \cup \{q\})$$

defines the transition map $\Theta^{\pm q} : A \times C^- \to \mathtt{1DML}(A)$. $\lhd$

The main technical result that we will prove is the following. Recall from Definition 7.16 that we write $\mathbb{S}, s_I \Vdash_s \mathbb{A}$ in case $\exists$ has a functional strategy in the game $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, s_I)$.

Proposition 8.23 *Let $\mathbb{A}$ be a disjunctive modal P-automaton, and let $\mathbb{S}$ be a Kripke model over some set $\mathsf{R} \supseteq \mathsf{P}$. Then the following are equivalent, for any state $s_I \in \mathbb{S}$:*
1) $\mathbb{S}, s_I \Vdash_s \widetilde{\exists} q.\mathbb{A}$;
2) $\mathbb{S}[q \mapsto Q], s_I \Vdash_s \mathbb{A}$, for some subset $Q \subseteq S$.

Proof. We only consider the case where $\mathsf{R} = \mathsf{P}$, leaving it for the reader to extend the result to the more general case (cf. Remark 8.21). Fix a disjunctive P-automaton $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$ and an R-model $\mathbb{S} = (S, R, V)$; to simplify notation we will write $c_t := \sigma_V(t)$, for an arbitrary point $t \in S$. Similarly, we will write $c - q := c \setminus \{q\}$ and $c + q := c \cup \{q\}$ for an arbitrary colour $c \in \wp(\mathsf{P})$. Furthermore, we will use the one-step presentation of the acceptance game, as in Table 11.

For the direction 1) $\Rightarrow$ 2) of the Proposition, assume that $\mathbb{S}, s_I \Vdash_s \widetilde{\exists} q.\mathbb{A}$. In other words, $\exists$ has a *functional* positional strategy f which is winning in the game $\mathcal{A}(\widetilde{\exists} q.\mathbb{A}, \mathbb{S})@(a_I, s_I)$. Abbreviate $\mathcal{A} := \mathcal{A}(\widetilde{\exists} q.\mathbb{A}, \mathbb{S})$.

Let $U \subseteq S$ be the set of points t in S such that, for some state $a \in A$, the position (a, t) is f-reachable in $\mathcal{A}@(a_I, s_I)$. It follows from functionality of f that for every $t \in U$ there is a *unique* such state in $\mathbb{A}$; we will denote this state as a_t. Furthermore, since f is a *winning strategy* in $\mathcal{A}@(a_I, s_I)$, every position of the form (a_t, t) is winning for $\exists$, and so by legitimacy of f, the marking $m_t : R[t] \to \wp(A)$ picked by f at this position is such that

$$(R[t], m_t) \Vdash^1 \Theta^{\pm q}(a_t, c_t). \tag{60}$$

Given that $\Theta^{\pm q}(a_t, c_t) = \Theta(a_t, c_t - q) \vee \Theta(a_t, c_t + q)$, this observation provides the set $Q \subseteq S$ that we are looking for:

$$Q := \{t \in U \mid (R[t], m_t) \Vdash^1 \Theta(a_t, c_t + q)\}.$$

We claim that $\mathbb{S}[q \mapsto Q], s_I \Vdash_s \mathbb{A}$, and to show this, we define the following positional strategy f_Q for $\exists$ in $\mathcal{A}_Q := \mathcal{A}(\mathbb{A}, \mathbb{S}[q \mapsto Q])$. At a position $(a, t) \in A \times S$, $\exists$ will play as follows:

- in case $t \in U$ and $a = a_t$, she picks the marking m_t;
- in all other cases she picks a random marking.

We first show that for each $t \in U$ and $a = a_t$ this strategy provides a legitimate move in $\mathcal{A}_Q$, that is,

$$(R[t], m_t) \Vdash^1 \Theta(a_t, \sigma_{V[q \mapsto Q]}(t)). \tag{61}$$

To see this, make the following case distinction:

- If $(R[t], m_t) \Vdash^1 \Theta(a_t, c_t + q)$ then by definition of Q we find $t \in Q$. This means that $\sigma_{V[q \mapsto Q]}(t) = \sigma_V(t) \cup \{q\} = c_t + q$. In other words, (61) holds indeed.
- If, on the other hand, $(R[t], m_t) \not\Vdash^1 \Theta(a_t, c_t + q)$ then by definition of Q we find $t \notin Q$. Furthermore, by (60) and the definition of $\Theta^{\pm q}$ it must be the case that $(R[t], m_t) \Vdash^1 \Theta(a_t, c_t - q)$. But since $t \notin Q$ we have $\sigma_{V[q \mapsto Q]}(t) = \sigma_V(t) \setminus \{q\} = c_t - q$, so that again we obtain (61).

It remains to show that f_Q is functional, and *winning* for $\exists$ in $\mathcal{A}_Q@(a_I, s_I)$, but this is in fact easy. The point is that at any position of the form (a_t, t) the strategies f and f_Q prescribe the *same* move, viz., m_t, and that at the position m_t the moves of $\forall$ in $\mathcal{A}$ and $\mathcal{A}_Q$ are the same. From this it follows that every position for $\exists$ that is reachable in an f_Q-guided match of $\mathcal{A}_Q@(a_I, s_I)$ is of the form (a_t, t) (with $t \in U$), and so by our previous claim about the legitimacy of f_Q at such positions, f_Q is a surviving strategy. Now consider an f_Q-guided full match of $\mathcal{A}_Q@(a_I, s_I)$; this very same match is also an f-guided match of $\mathcal{A}$, and hence won by $\exists$ — after all we assumed that f is a winning strategy for $\exists$ in $\mathcal{A}(a_I, s_I)@(a_I, s_I)$, and the winning conditions in $\mathcal{A}_Q$ and $\mathcal{A}$ are the same. In other words, every f_Q-guided full match of $\mathcal{A}_Q@(a_I, s_I)$ is won by $\exists$. Finally, since f is a functional strategy, so is f_Q. This finishes the proof that 1) $\Rightarrow$ 2).

The proof of the opposite implication, 2) $\Rightarrow$ 1), is similar; we omit the details. QED

From this, Theorem 8.20 is almost immediate.

Proof of Theorem 8.20. Let P and q be a set of proposition letters and a proposition letter, respectively, let $\mathbb{A}$ be a disjunctive modal automaton over P, and let $(\mathbb{S}, r)$ be a pointed model over a set R of proposition letters such that $\mathsf{P} \subseteq \mathsf{R}$. It suffices to show that

$$\mathbb{S}, r \Vdash \widetilde{\exists} q.\mathbb{A} \text{ iff } \mathbb{S}', r' \Vdash \mathbb{A}, \text{ for some } (\mathbb{S}', r') \text{ with } \mathbb{S}, r \leftrightarroweq_{\mathsf{R}\setminus q} \mathbb{S}', r'. \tag{62}$$

But since $\mathbb{A}$ is disjunctive, it is easy to see that $\widetilde{\exists} q.\mathbb{A}$ is disjunctive as well, and so it follows from Theorem 7.18 that

$$\mathbb{S}, r \Vdash \widetilde{\exists} q.\mathbb{A} \text{ iff } \mathbb{S}', r' \Vdash_s \widetilde{\exists} q.\mathbb{A}, \text{ for some } (\mathbb{S}', r') \text{ with } \mathbb{S}, r \leftrightarroweq_{\mathsf{R}\setminus q} \mathbb{S}', r'. \tag{63}$$

Combining this with Proposition 8.23 we find

$$\mathbb{S}, r \Vdash \widetilde{\exists} q.\mathbb{A} \text{ iff } \mathbb{S}'[q \mapsto Q], r' \Vdash_s \mathbb{A}, \text{ for some } (\mathbb{S}', r') \text{ with } \mathbb{S}, r \leftrightarroweq_{\mathsf{R}\setminus q} \mathbb{S}', r' \text{ and some } Q \subseteq S'. \tag{64}$$

Now it is obvious that $\mathbb{S}'[q \mapsto Q], r' \leftrightarroweq_{\mathsf{R}\setminus q} \mathbb{S}', r'$. But then (62) is immediate. QED

Finishing this section, we show how to derive the uniform interpolation property from the definability of the bisimulation quantifiers.

Proof of Theorem 8.17. Fix the formula φ and the set P, and let $q_1, \ldots, q_n$ enumerate the free variables of φ that are *not* in P, that is, $\{q_1, \ldots, q_n\} = FV(\varphi) \setminus \mathsf{P}$. We claim that the formula $\widetilde{\exists} q_1 \cdots \widetilde{\exists} q_n.\varphi$ is the required (right) uniform interpolant of φ with respect to P.

To prove this, take an arbitrary formula ψ such that $FV(\psi) \cap FV(\varphi) \subseteq \mathsf{P}$. Clearly this implies that no q_i is a free variable of ψ. We first show that

$$\varphi \models \widetilde{\exists} q_1 \cdots \widetilde{\exists} q_n.\varphi.$$

To see this, let $(\mathbb{S}, s)$ be some pointed Kripke model (over some set $\mathsf{R} \supseteq FV(\varphi)$) such that $\mathbb{S}, s \Vdash \varphi$. Since we obviously have that $\mathbb{S}, s \leftrightarroweq_{\mathsf{R}\setminus q} \mathbb{S}, s$ for *any* proposition letter q, it easily follows that $\varphi \models \widetilde{\exists} q_1 \cdots \widetilde{\exists} q_n.\varphi$. This takes care of the right-to-left direction from (57).

For the opposite direction of (57), assume that $\varphi \models \psi$, and let $(\mathbb{S}, s)$ be a pointed Kripke model such that $\mathbb{S}, s \Vdash \widetilde{\exists} q_1 \cdots \widetilde{\exists} q_n.\varphi$. It follows that there is a sequence $(\mathbb{S}_i, s_i)_{0 \leq i \leq n}$ of pointed models such that $(\mathbb{S}, s) = (\mathbb{S}_0, s_0)$, $\mathbb{S}_n, s_n \Vdash \varphi$, and $\mathbb{S}_i, s_i \leftrightarroweq_{\mathsf{R}\setminus q_{i+1}} \mathbb{S}_{i+1}, s_{i+1}$ for all i with $0 \leq i < n$. Then by assumption it follows from $\mathbb{S}_n, s_n \Vdash \varphi$ that $\mathbb{S}_n, s_n \Vdash \psi$. But since none of the proposition letters q_i is free in ψ, step by step applying the bisimulation invariance of the modal μ-calculus we may show that each pointed model $\mathbb{S}_i, s_i$ satisfies ψ. In particular, we find that $\mathbb{S}, s \Vdash \psi$, as required. QED

Notes

The decidability of the satisfiability problem of the modal μ-calculus was first proved by Kozen and Parikh [17] via a reduction to *SnS*. Emerson & Jutla [10] established the EXPTIME-completeness of this problem. The finite model property was proved by Kozen [16].

Uniform interpolation of the modal μ-calculus was proved by D'Agostino & Hollenberg [8], who established some other model-theoretic results as well.

Exercises

Exercise 8.1 Let γ be some disjunctive fixed point formula.

(a) Show that $\mu x.\gamma$ is satisfiable iff $\gamma[\bot/x]$ is satisfiable.

(b) Show that $\nu x.\gamma$ is satisfiable iff $\gamma[\top/x]$ is satisfiable.

(c) Do the above statements hold for arbitrary fixed point formulas as well?

Exercise 8.2 Prove the left-to-right direction of (72) in Proposition 9.28.

Exercise 8.3 Is disjunctivity of the automaton $\mathbb{A}$ needed in the proof of Proposition 8.23?

Exercise 8.4 (PDL + bisimulation quantifier) Consider a setting with finitely many atomic actions. Let PDL+$\widetilde{\exists}$ be the extension of propositional dynamic logic with (explicit) bisimulation quantifiers. Show that there is a (truth-preserving) translation from the modal μ-calculus to PDL+$\widetilde{\exists}$.

9 Expressive completeness

In this chapter we compare the expressive power of the modal μ-calculus to that of monadic second-order logic. The key result that we will prove is that the modal μ-calculus has the same expressive power as the bisimulation invariant fragment of monadic second-order logic, in brief:

$$\mu\mathtt{ML} \equiv \mathtt{MSO}/{\underline{\leftrightarrow}}. \tag{65}$$

In fact, Theorem 9.21, the actual result that we are going to prove is a bit stronger than (65).

Our proof will be automata-theoretic in nature: after discussing two different (but equivalent) versions of monadic second-order logic in section 9.1, we show in section 9.2 that on tree models, *MSO* has the same expressive power as the class $Aut(\mathtt{1FOE})$ of automata over the one-step logic 1FOE. Since the modal μ-calculus corresponds to the class $Aut(\mathtt{1FO})$, we will prove (65) in section 9.3 via a comparison of the one-step languages 1FOE and 1FO.

9.1 Monadic second-order logic

Second-order logic is the extension of first-order logic where quantification is allowed, not only over individuals, but also over relations on the domain. In *monadic* second-order logic, this second-order quantification is restricted to unary relations, that is, subsets of the domain. The syntax of monadic second-order logic is usually defined as the extension of that of first-order logic by second-order quantifiers of the form $\exists p/\forall p$, where p is a monadic predicate symbol.

Definition 9.1 Given a set D of atomic actions, a set IVar of individual variables and a set Prop of set variables, we define the language $\mathtt{MSO}^2_{\mathsf{D}}$ as follows:

$$\varphi \ ::= \ x \doteq y \ \mid \ R_d xy \ \mid \ p(x) \ \mid \ \neg\varphi \ \mid \ \varphi \vee \varphi \ \mid \ \exists x.\varphi \ \mid \ \exists p.\varphi$$

Here x and y are variables from IVar, p is a variable from P, and $d \in \mathsf{D}$ is an atomic action.

We let $\mathtt{MSO}^2_{\mathsf{D}}(\mathsf{X},\mathsf{P})$ denote the set of $\mathtt{MSO}^2_{\mathsf{D}}$-formulas φ of which all individual free variables are from X and all free set variables are from P. In case X is a singleton $\{x\}$, we write $\mathtt{MSO}^2_{\mathsf{D}}(x,\mathsf{P})$ rather than $\mathtt{MSO}^2_{\mathsf{D}}(\{x\},\mathsf{P})$ $\lhd$

This semantics of this language is completely standard, with $\exists x$ denoting first-order quantification (that is, quantification over individual states), and $\exists p$ denoting monadic second-order quantification (that is, quantification over sets of states).

It turns out, however, that for a nice inductive translation of *MSO* to automata, it is more convenient to use a slightly nonstandard version of *MSO* that is *single-sorted* in that it only admits second-order variables, not first-order ones. Quantification over individuals can then be simulated by quantification over singleton sets. In addition, to facilitate the comparison with modal languages, which are interpreted in *pointed* Kripke models, we need to install a feature in the language that allows access to the designated or actual world of the Kripke model.

Definition 9.2 Given a set D of atomic actions, we define the language of *monadic second-order logic* $\mathtt{MSO}_{\mathsf{D}}$ as follows:

$$\varphi ::= p \sqsubseteq q \mid R_d pq \mid \Downarrow p \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists p.\varphi,$$

where p and q are propositional variables from P. We let $\mathtt{MSO}_{\mathsf{D}}(\mathsf{P})$ denote the set of $\mathtt{MSO}_{\mathsf{D}}$-formulas of which the free variables are from P. $\lhd$

Definition 9.3 Given a Kripke model $\mathbb{S} = \langle S, V, R\rangle$, and a designated point $s \in S$, we define the semantics of $\mathtt{MSO}$ as follows:

$$\begin{array}{lll}
\mathbb{S}, s \models p \sqsubseteq q & \text{if} & V(p) \subseteq V(q) \\
\mathbb{S}, s \models R_d pq & \text{if} & \text{for all } t \in V(p) \text{ there is a } u \in V(q) \text{ with } R_d tu \\
\mathbb{S}, s \models \Downarrow p & \text{if} & V(p) = \{s\} \\
\mathbb{S}, s \models \neg\varphi & \text{if} & \mathbb{S}, s \not\models \varphi \\
\mathbb{S}, s \models \varphi \vee \psi & \text{if} & \mathbb{S}, s \models \varphi \text{ or } \mathbb{S}, s \models \psi \\
\mathbb{S}, s \models \exists p.\varphi & \text{if} & \mathbb{S}[p \mapsto X], s \models \varphi \text{ for some } X \subseteq S.
\end{array}$$

An $\mathtt{MSO}$-formula φ is *bisimulation invariant* if $\mathbb{S}, s \leftrightarroweq \mathbb{S}', s'$ implies that $\mathbb{S}, s \models \varphi \Leftrightarrow \mathbb{S}', s \models \varphi$. $\lhd$

Remark 9.4 In fact, one may think of the formalism as a *first-order* logic of which the intended models are *power structures* of the form $\langle \wp(S), \subseteq, \vec{R}, \{s\}\rangle$, where $R_d(Y, Z)$ iff for all $y \in Y$ there is a $z \in Z$ such that $R_d yz$.) $\lhd$

It is not too hard to see that the two languages are in fact equivalent.

Theorem 9.5 *There are effective procedures transforming a formula in* $\mathtt{MSO}^2(x, \mathsf{P})$ *into an equivalent* $\mathtt{MSO}(\mathsf{P})$*-formula, and vice versa:*

$$\mathtt{MSO}^2 \equiv \mathtt{MSO}.$$

To start with, there is a straightforward, inductively defined translation $(\cdot)' : \mathtt{MSO}_{\mathsf{D}}(\mathsf{P}) \to \mathtt{MSO}^2_{\mathsf{D}}(x, \mathsf{P})$ such that

$$\mathbb{S}, s \models \varphi \text{ iff } \mathbb{S} \models \varphi'[s],$$

for all formulas $\varphi \in \mathtt{MSO}_{\mathsf{D}}(\mathsf{P})$ and all pointed Kripke models $\mathbb{S}$. The only interesting clause in the inductive definition of this translation concerns the $\Downarrow$-connective, for which we set

$$(\Downarrow p)' := \forall y(p(y) \leftrightarrow y \doteq x).$$

For the opposite direction, the key observation is that $\mathtt{MSO}$ can interpret $\mathtt{MSO}^2$ by encoding individual variables as set variables denoting *singletons*. To understand how this works, we need to have a closer look at the semantics. Formulas of the language $\mathtt{MSO}^2$ are interpreted over Kripke models $\mathbb{S}$ with an *assignment*, that is, a map $\alpha : \mathsf{IVar} \to S$ interpreting the individual variables as elements of S. But then we can encode such an $\mathtt{MSO}^2$-model $\mathbb{S} = (S, R, V)$ with assignment α, as the $\mathtt{MSO}$-model $\mathbb{S}^\alpha := (M, R, V^\alpha)$ over $\mathsf{Prop} \cup \mathsf{IVar}$, where $V^\alpha(p) := V(p)$ if p is a set variabe, and $V^\alpha(x) := \{\alpha(x)\}$ if x is an individual variable.

Proposition 9.6 *There is a translation* $(\cdot)^t : \mathtt{MSO}^2_{\mathsf{D}}(\mathsf{X},\mathsf{P}) \to \mathtt{MSO}_{\mathsf{D}}(\mathsf{P} \uplus \mathsf{X})$ *such that*

$$\mathbb{S} \models \varphi[\alpha] \textit{ iff } \mathbb{S}^{\alpha} \models \varphi^t \tag{66}$$

for all $\varphi \in \mathtt{MSO}^2_{\mathsf{D}}(\mathsf{X},\mathsf{P})$*, all Kripke models* $\mathbb{S} = (S,R,V)$ *and all assignments* $\alpha : \mathsf{X} \to S$*.*

As a corollary, for all $\varphi \in \mathtt{MSO}^2_{\mathsf{D}}(x,\mathsf{P})$ *and all pointed Kripke models* $(\mathbb{S},s)$ *we obtain*

$$\mathbb{S} \models \varphi[s] \textit{ iff } \mathbb{S}, s \models \forall x.(\Downarrow x \to \varphi^t). \tag{67}$$

Proof. The translation crucially involves the MSO-formulas $\mathtt{empty}(p)$ and $\mathtt{sing}(p)$ given by

$$\begin{array}{lcl} \mathtt{empty}(p) & := & \forall q\,(p \sqsubseteq q) \\ \mathtt{sing}(p) & := & \forall q\,\big(q \sqsubseteq p \to (\mathtt{empty}(q) \vee p \sqsubseteq q)\big). \end{array}$$

It is not hard to prove that these formulas hold in $\mathbb{S}$ iff, respectively, $V(p)$ is empty and $V(p)$ is a singleton.

With these formulas defined, we can now inductively fix the translation as follows:

$$\begin{array}{lcl} (p(x))^t & := & x \sqsubseteq p \\ (R_d xy)^t & := & R_d xy \\ (x \doteq y)^t & := & x \sqsubseteq y \wedge y \sqsubseteq x \\ (\neg\varphi^t & := & \neg\varphi^t \\ (\varphi_0 \vee \varphi_1)^t & := & \varphi_0^t \vee \varphi_1^t \\ (\exists x.\varphi)^t & := & \exists x.(\mathtt{sing}(x) \wedge \varphi^t) \\ (\exists p.\varphi)^t & := & \exists p.\varphi^t \end{array}$$

It is a routine exercise to verify (66), so we leave the details for the reader. Similarly, the proof of (67) is immediate by (66) and the definitions of the semantics of $\Downarrow$. QED

Note that the translation $(\cdot)^t$ given in the proof of Proposition 9.6 does not involve the connective $\Downarrow$. The only use of $\Downarrow$ in this setting is to mark the designated node of a *pointed* Kripke model.

9.2 Automata for monadic second-order logic

The aim of this section is to provide an automata-theoretic perspective on monadic second-order logic. That is, we will provide a construction transforming an arbitrary MSO-formula φ into an automaton $\mathbb{B}_{\varphi}$ that is equivalent to φ, at least, if we confine attention to tree models. In fact, we will encounter various kinds of automata, all corresponding to MSO-formulas, and all taking some fragment of *monadic first-order logic* as the co-domain of their transition map, as in Definition 7.26 and Definition 7.27.

Recall that the set $\mathtt{MFOE}(A)$ of *monadic first-order formulas over* A is given by the following grammar:

$$\alpha \;::=\; \top \mid \bot \mid a(x) \mid \neg a(x) \mid x \doteq y \mid x \not\doteq y \mid \alpha \vee \alpha \mid \alpha \wedge \alpha \mid \exists x.\alpha \mid \forall x.\alpha$$

where $a \in A$ and x, y are first-order (individual) variables, and that $\mathtt{MFO}(A)$ is the set of $\mathtt{MFOE}(A)$-formulas without occurrences of identity formulas (or their negations). Recall as

Position	Player	Admissible moves
$(a,s) \in A \times S$	$\exists$	$\{U : A \to \wp(R(s)) \mid (R(s), U) \models \Theta(a, \sigma_V(s))\}$
$U : A \to \wp(S)$	$\forall$	$\{(b,t) \mid t \in U(b)\}$

Table 16: Acceptance game for *MSO*-automata

well that $\mathtt{1FOE}(A)$ and $\mathtt{1FO}(A)$ are the one-step languages consisting of the sentences of, respectively, $\mathtt{MFOE}(A)$ and $\mathtt{MFO}(A)$, where each monadic predicate $a \in A$ occurs only positively. It will be convenient in this section to present one-step models using valuations rather than markings; that is, a *one-step model* will be denoted as a pair (Y, V) consisting of some set Y and an A-valuation $V : A \to \wp(Y)$.

Definition 9.7 An *MSO-automaton* over a set P of proposition letters is nothing but a $\mathtt{1FOE}$-automaton over P, that is, a quadruple $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$, where A, a_I and Ω are as usual, and Θ is a map $\Theta : A \times \wp(\mathsf{P}) \to \mathtt{1FOE}(A)$.

The acceptance game of such an automaton with respect to a Kripke model $\mathbb{S}$ is given in Table 9.2. The winning conditions for both finite and infinite matches are as usual. $\lhd$

In words, the acceptance game proceeds as follows. At a basic position (a, s), $\exists$ chooses a valuation U interpreting each 'predicate' $a \in A$ as a subset $U(a)$ of the set $R(s)$ of successors of s. In this choice, she is bound by the condition that the sentence $\Theta(a, \sigma_V(s))$ must be *true* in the resulting A-structure $(R(s), U)$. Once chosen, this map U itself determines the next position of the match. As a position, U belongs to player $\forall$, and all he has to do is to choose a pair (b, t) such that $t \in U(b)$. This pair (b, t) is then the next basic position of the match.

The link with modal automata is given by Proposition 7.31, stating that, seen as one-step languages, $\mathtt{1FO}$ is equivalent to $\mathtt{1ML}$. From this we obtain the equivalence in expressive power of the automata classes $Aut(\mathtt{1FO})$ and $Aut(\mathtt{1ML})$, which in its turn entails the following.

Theorem 9.8 *There are effective procedures transforming a μ-calculus formula into an equivalent MSO-automaton in $Aut(\mathtt{1FO})$, and vice versa:*

$$\mu\mathtt{ML} \equiv Aut(\mathtt{1FO}).$$

The main result of this section states a very similar result for $\mathtt{MSO}$ and arbitrary *MSO*-automata, if we confine our attention to *tree models*:

Theorem 9.9 *There are effective procedures transforming an* $\mathtt{MSO}$*-formula ξ into an MSO-automaton $\mathbb{A}$, and vice versa, such that the corresponding formula ξ and automaton $\mathbb{A}$ are equivalent on the class of tree models:*

$$\mathtt{MSO} \equiv Aut(\mathtt{1FOE}) \text{ (on tree models).}$$

Note that on *arbitrary* models, monadic second-order logic can express properties that cannot be captured by *MSO*-automata. For instance, it is easy to write an $\mathtt{MSO}$-formula stating that the designated point of a Kripke model lies on a cycle, but there is no *MSO*-automaton that recognizes exactly the class of pointed Kripke models with this property.

We will prove the two directions in the statement of Theorem 9.9 separately. Leaving the transformation of automata to monadic second-order formulas to the end of the section, we first concentrate on the opposite direction.

Proposition 9.10 *There is an effective procedure transforming a formula $\varphi \in \mathtt{MSO}(\mathsf{P})$ into an MSO-automaton $\mathbb{B}_\varphi$ over P that is equivalent to φ over the class of tree models. That is:*

$$\mathbb{S}, r \models \varphi \text{ iff } \mathbb{B}_\varphi \text{ accepts } (\mathbb{S}, r). \tag{68}$$

for any tree model $\mathbb{S}$ with root r.

We will prove Proposition 9.10 by induction on the complexity of *MSO*-formulas. The proposition below takes care of the atomic case.

Proposition 9.11 *Let φ be one of the atomic MSO-formulas: Rpq, $p \sqsubseteq q$, or $\Downarrow p$. Then there is an MSO-automaton $\mathbb{B}_\varphi$ that is equivalent to φ on tree models.*

Proof. We restrict attention to the formula Rpq, leaving the other cases as an exercise for the reader. The automaton $\mathbb{B}_{Rpq}$ is defined as the structure $(\{a_0, a_1\}, \Theta, \Omega, a_0)$, where Θ is given by putting:

$$\begin{array}{rcl} \Theta(a_0, c) & := & \begin{cases} \exists y \left(a_1(y) \land \forall z\, (z \neq y \to a_0(z))\right) & \text{if } p \in c \\ \forall z\, a_0(z) & \text{otherwise} \end{cases} \\ \Theta(a_1, c) & := & \begin{cases} \bot & \text{if } q \notin c \\ \exists y \left(a_1(y) \land \forall z\, (z \neq y \to a_0(z))\right) & \text{if } q \in c \text{ and } p \in c \\ \forall z\, a_0(z) & \text{otherwise} \end{cases} \end{array}$$

Furthermore, Ω is defined via $\Omega(a_i) := 0$ for each a_i — as a consequence, $\exists$ wins all infinite games. We leave it for the reader to verify that this automaton is of the right shape, and that it is indeed equivalent to the formula Rpq on tree models. QED

For the inductive step of the argument, there are three cases to consider, corresponding to, respectively, the connectives $\vee$ and $\neg$, and the (second-order) existential quantification. It turns out that the first two cases are relatively easy to handle, cf. Proposition 7.38. To take care of the existential quantification however, we need to work with *nondeterministic* automata, in which every formula $\Theta(a, c)$ has been brought into a certain normal form. Fortunately, we can prove a simulation theorem for *MSO*-automata, implying that we may transform any *MSO*-automaton into an equivalent nondeterministic one. We need some definitions on these normal forms of `1FOE`-formulas.

Definition 9.12 Fix a set A of propositional variables. We introduce some abbreviations for `MFOE`-formulas:

$$\mathtt{diff}(y_1, \ldots, y_n) := \bigwedge\{y_i \neq y_j \mid 1 \leq i < j \leq n\},$$

and, for a set $B \subseteq A$:

$$\tau_B(x) := \bigwedge_{a \in B} a(x).$$

Now define the following MFOE-sentences:

$$\begin{array}{rcl} \chi^{=}_{\overline{B},\overline{C}} & := & \exists y_1 \cdots y_n \Big(\mathtt{diff}(\overline{y}) \wedge \bigwedge_i \tau_{B_i}(y_i) \wedge \forall z\, (\mathtt{diff}(\overline{y}, z) \to \bigvee_j \tau_{C_j}(z)) \Big) \\ \chi_{\overline{B},\overline{C}} & := & \exists y_1 \cdots y_n \Big(\bigwedge_i \tau_{B_i}(y_i) \wedge \forall z \bigvee_j \tau_{C_j}(z) \Big) \end{array}$$

where $\overline{B} = B_1, \ldots, B_n$ and $\overline{C} = C_1, \ldots, C_m$ are two sequences of subsets of A.

Sentences of the form $\chi^{=}(\overline{B}, \overline{C})$ are said to be in *basic form*, and in *special basic form* in case each B_i and C_j is a singleton. The sets of these formulas are denoted as $\mathtt{BF}(A)$ and $\mathtt{SBF}(A)$, respectively. ◁

In words, the formula $\mathtt{diff}(y_1, \ldots, y_n)$ expresses that the variables $y_1, \ldots, y_n$ refer to n *distinct* objects of the domain. The formula $\tau_B(x)$ can be seen to state that x *realises* the type B, that is: it satisfies all predicates a in B. The formula $\chi^{=}_{\overline{B},\overline{C}}$ expresses the existence of n distinct objects realising the B-types, with all other objects realising one of the C-types. This formula is (equivalent to) the formula $\forall z \bigvee_j \tau_{C_j}(z)$ in the special case where $n = 0$, and, in case $m = 0$ as well, to the formula $\forall z \perp$ (which holds in the empty model only). As a simplified version of $\chi^{=}_{\overline{B},\overline{C}}$, the sentence $\chi_{\overline{B},\overline{C}}$ states that all types in $\overline{B}$ are witnessed by some object, while every object satisfies some C-type. Note that for the latter reason, $\chi_{\overline{B},\overline{C}}$ is generally not a semantics consequence of $\chi^{=}_{\overline{B},\overline{C}}$. Finally, observe that $\chi^{=}_{\overline{B},\overline{C}}$ and $\chi_{\overline{B},\overline{C}}$ are positive sentences, and hence, one-step formulas in 1FOE and 1FO, respectively.

Using these normal forms, we can now define the notion of a nondeterministic *MSO*-automaton.

Definition 9.13 An *MSO*-automaton $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$ is called *nondeterministic* if $\mathsf{Ran}(\Theta) \subseteq \mathtt{Dis}(\mathtt{SBF}(A))$, that is, every formula $\Theta(a, c)$ is a disjunction of special basic formulas. ◁

Nondeterministic automata are of interest because they admit *functional strategies* — in tree models, that is. As in Definition 7.16, we call a strategy f for $\exists$ in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, r)$ *functional* if for every $s \in S$ there is at most one $a \in A$ such that the position (a, s) is reachable in an f-guided match of $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, r)$. In case $\exists$ has a functional strategy which is in addition winning, we write $\mathbb{S}, r \Vdash_s \mathbb{A}$. The following proposition states that on tree models, we may *always* assume that winning strategies are functional.

Proposition 9.14 *Let $\mathbb{A}$ be a nondeterministic MSO-automaton, and let $\mathbb{S}$ be a tree-based Kripke model with root r. Then $\mathbb{S}, r \Vdash \mathbb{A}$ iff $\mathbb{S}, r \Vdash_s \mathbb{A}$.*

As a corollary, nondeterministic *MSO*-automata are closed under existential second-order quantification.

Corollary 9.15 *Let $\mathbb{D} = \langle D, \Delta, \Omega, d_I \rangle$ be a nondeterministic MSO-automaton over the set $\mathsf{P} \cup \{p\}$. Then there is a nondeterministic automaton $\mathbb{D}^{\exists p}$ over P, such that for all tree models $(\mathbb{S}, r)$:*

$$\mathbb{D}^{\exists p} \text{ accepts } (\mathbb{S}, r) \text{ iff } \mathbb{D} \text{ accepts } (\mathbb{S}[p \mapsto T], r) \text{ for some } T \subseteq S. \tag{69}$$

Proof. Define the automaton $\mathbb{D}^{\exists p} := \langle D, \Delta^{\exists p}, \Omega, d_I \rangle$, with alphabet $C = \wp(\mathsf{P})$, by putting

$$\Delta^{\exists p}(a, c) := \Delta(a, c) \vee \Delta(a, c \cup \{p\}).$$

Clearly then $\mathbb{D}^{\exists p}$ is a nondeterministic *MSO*-automaton, so it remains to prove that $\mathbb{D}^{\exists p}$ satisfies (69). But since we may assume winning strategies to be functional, this proof is a variation on a proof given earlier, viz., that of Proposition 8.23. QED

But if *nondeterministic MSO*-automata admit existential second-order quantification, in order to transfer this closure property to the class of arbitrary automata, all we need is the following Simulation Theorem which states in particular that every *MSO*-automaton has a nondeterministic equivalent.

Theorem 9.16 (Simulation Theorem) *There are effective constructions transforming an automaton of any of the kinds below to an equivalent automaton of any other kind:*

1) $Aut(\mathtt{1FOE})$*,*

2) $Aut(\mathtt{Dis}(\mathtt{BF}(A)))$*,*

3) $Aut(\mathtt{Dis}(\mathtt{SBF}(A)))$*.*

To prove the implication from 1) to 2) of this result, we need a model-theoretic result on monadic first-order logic, that will be of use later on as well.

Proposition 9.17 *There is an effective procedure transforming an arbitrary positive sentence in* $\mathtt{MFOE}(A)$ *to an equivalent disjunction of sentences in basic form.*

The proof of this result, which we omit for the time being, is a fairly straightforward exercise in the theory of Ehrenfeucht-Fraïssé games.

Proof of Theorem 9.16. The implications from 3) to 2) and from 2) to 1) are trivial consequences of the definitions. The implication from 1) to 2) is immediate by Proposition 9.17.

The hardest part of the proof concerns the remaining implication, from 2) to 3). This, however, is an instance of the general simulation theorem that we proved in section 7.6. We only need to verify that the language $\mathtt{Dis}(\mathtt{SBF})$, seen as a one-step language, is $\bigwedge$-distributive over $\mathtt{Dis}(\mathtt{BF})$, and therefore, over $\mathtt{1FOE}$, but we leave this as an exercise for the reader. QED

With this Simulation Theorem we have all the results that are needed for the inductive translation of second-order formulas to $\mathtt{MSO}$-automata.

Proof of Proposition 9.10. As mentioned, the proposition is proved by induction on the complexity of $\varphi \in \mathtt{MSO}$. The atomic case of the induction is covered by Proposition 9.11. For the induction step, the case where $\varphi = \exists p.\psi$ is taken care of by Theorem 9.16 and Corollary 9.15. The remaining cases, where respectively $\varphi = \neg\psi$ and $\varphi = \varphi_0 \vee \varphi_1$, are left as exercises for the reader. QED

Proposition 9.10 takes care of one direction of Theorem 9.9; for the opposite direction we need to find an equivalent formula $\xi_{\mathbb{A}} \in \mathtt{MSO}$ for each *MSO*-automaton $\mathbb{A}$.

Proposition 9.18 *There is an effective procedure transforming an MSO-automaton $\mathbb{A}$ into a formula $\xi_{\mathbb{A}} \in \mathtt{MSO}^2(\mathsf{P})$ that is equivalent to φ over the class of tree models. That is:*

$$\mathbb{A} \textit{ accepts } (\mathbb{S}, r) \textit{ iff } \mathbb{S}, r \models \xi_{\mathbb{A}}. \tag{70}$$

for any tree model $\mathbb{S}$ with root r.

Proof. For the time being we confine ourselves to a proof sketch. The basic idea is to encode the operational semantics of an *MSO*-automaton in monadic second-order logic; this works for nondeterministic automata over tree models, since we can express the working of a functional strategy.

To give a bit more detail, fix an *MSO*-automaton $\mathbb{A}$. We first transform $\mathbb{A}$ into an equivalent nondeterministic automaton $\mathbb{D} = (D, \Theta, \Omega, d_I)$; this is possible by Theorem 9.16. It then suffices to write down a monadic second-order formula $\xi(x)$ in $\mathtt{MSO}^2(x)$ such that, for an arbitrary tree model $\mathbb{S}$ with root r:

$$\mathbb{S} \models \xi[r] \text{ iff } \exists \text{ has a functional positional winning strategy in } \mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, r).$$

Let $\mathbb{S} = (S, R, V)$ be a an arbitrary tree model with root r and let $D = \{a_1, \ldots, a_n\}$. Here we think of the a_i as second-order variables that will be quantified over existentially, in order to express the *existence* of a functional positional strategy. Take an arbitrary valuation $U : D \to \wp(S)$. It is easy to write down an $\mathtt{MSO}^2(x)$-formula $\varphi(\overline{a}, x)$ which holds of the resulting model $\mathbb{S} \oplus U$ iff $|U(a_i)| \leq 1$ for each i, so that we may think of the associated marking m_U as a potential functional strategy of $\exists$ in the acceptance game $\mathcal{A}(\mathbb{A}, \mathbb{S})$. Writing a_s for the unique state such that $a_s \in m_U$, we may then use the one-step formula $\Theta(a, \sigma_V(s))$ as a basis for a first-order formula which expresses that this potential strategy induced by U actually provides a legitimate move for $\exists$ at position (a_s, s). Finally, note that any infinite match of $\mathcal{A}(\mathbb{A}, \mathbb{S})@(a_I, r)$ corresponds to a branch of $\mathbb{S}$ (that is, an infinite path starting at r); using a second-order variable b to range over such branches, it is then fairly straightforward to write down a formula stating that the highest parity occurring infinitely often on any match of an m_U-guided match is even. QED

9.3 Expressive completeness modulo bisimilarity

A central result in the theory of basic modal logic states that modal logic corresponds to the bisimulation invariant fragment of first-order logic. In this section we will prove an extension of this result stating that the modal μ-calculus is the bisimulation invariant fragment of monadic *second-order* logic. While it is not difficult to show that every $\mu\mathtt{ML}$-formula is equivalent to a bisimulation-invariant formula in $\mathtt{MSO}$, it is the converse correspondence where the true importance of the result lies. We may see it as an expressive completeness result, stating that the modal μ-calculus is sufficiently strong to express *every* bisimulation-invariant formula in monadic second-order logic. Note that in a context such as process theory, where we consider bisimilar pointed Kripke models as different representations of the *same* process, bisimulation-invariant properties are in fact the *only* relevant ones. In such a situation, we may read the bisimulation-invariance result as saying that modal fixpoint logic has the same expressive power as monadic second-order logic, when it comes to expressing relevant properties.

► `Add examples of what can be expressed in MSO, and not in` μ`ML:`
`- every point has exactly two d-successors`
`- the actual state does not lie on a cycle`

We first show that there is truth-preserving translation mapping every formula of the modal μ-calculus to an equivalent monadic second-order formula. Recall from Remark 9.4 that $\mathtt{MSO}^2_{\mathsf{D}}(x, \mathsf{P})$ is the standard (two-sorted) version of monadic second-order logic.

Definition 9.19 For any individual variable x we define, by induction on the complexity of a formula $\varphi \in \mu\mathtt{ML}_{\mathsf{D}}$, a translation $\mathtt{ST}_x : \mu\mathtt{ML}_{\mathsf{D}}(\mathsf{P}) \to \mathtt{MSO}^2_{\mathsf{D}}(x, \mathsf{P})$.

$$\begin{array}{lll}
\mathtt{ST}_x(p) & := & p(x) \\
\mathtt{ST}_x(\neg\varphi) & := & \neg\mathtt{ST}_x(\varphi) \\
STx(\Diamond_d\varphi) & := & \exists y(R_d xy \land \mathtt{ST}_y(\varphi)) \\
STx(\Diamond\varphi) & := & \exists y(Rxy \land \mathtt{ST}_y(\varphi)) \\
STx(\mu p.\varphi) & := & \exists p.\Big(p(x) \land \forall y.\big(p(y) \leftrightarrow \forall q.(\mathtt{PRE}(\varphi, q) \to q(y))\big)\Big),
\end{array}$$

where $\mathtt{PRE}(\varphi, q)$ abbreviates the formula $\forall y.(\mathtt{ST}_y(\varphi)[q/p] \to q(y))$. ◁

Theorem 9.20 *For any formula $\varphi \in \mu\mathtt{ML}$ we have $\varphi \equiv \mathtt{ST}_x(\varphi)$, in the sense that*

$$\mathbb{S}, s \Vdash \varphi \textit{ iff } \mathbb{S} \models \mathtt{ST}_x(\varphi)[s]$$

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

Proof. The proof of this theorem can be proved by a straightforward induction on the complexity of $\mu\mathtt{ML}$-formulas.

For the inductive clause of the least fixpoint operator μ, consider the formula $\mu x.\varphi$. We leave it for the reader to verify (using the inductive hypothesis) that the formula $\mathtt{PRE}(\varphi, q)$ expresses that q is a pre-fixpoint of φ, and that the formula $\forall y.\big(p(y) \leftrightarrow \forall q.(\mathtt{PRE}(\varphi, q) \to q(y))\big)$ expresses that p is the intersection of all pre-fixpoints of φ. QED

In the other direction, the actual result that we will prove is somewhat stronger than mere expressive completeness.

Theorem 9.21 *There is an effectively defined translation $(\cdot)^* : \mathtt{MSO} \to \mu\mathtt{ML}$ such that a formula $\varphi \in \mathtt{MSO}$ is invariant under bisimulations iff it is equivalent to φ^*.*

We will prove this result by automata-theoretic means. Recall that in the previous section we obtained the following characterisations of the languages MSO and μML:

$$\begin{array}{lll}
\mathtt{MSO} & \sim & Aut(\mathtt{1FOE}) \quad \text{(on trees)} \\
\mu\mathtt{ML} & \sim & Aut(\mathtt{1FO}).
\end{array}$$

The translation $(\cdot)^* : \mathtt{MSO} \to \mu\mathtt{ML}$ mentioned in Theorem 9.21 will be based on a construction transforming 1FOE-automata into 1FO-automata, whereas this construction in its turn is based on a translation $(\cdot)^*$ at the one-step level. For the details, we need to develop some rudimentary model theory at the level of monadic first-order logic, in this case linking the one-step languages MFOE and MFO.

Recall from Definition 7.26 that $\mathtt{1FOE}(A)$ and $\mathtt{1FO}(A)$ denote the sets of A-positive sentence in the languages $\mathtt{MFOE}(A)$ and $\mathtt{MFO}(A)$ of monadic first-order logic with and without identity, respectively. Our translation $(\cdot)^*$ involves the *basic forms* of Definition 9.13. Based on Proposition 9.17, we can provide the required translation from 1FOE to 1FO.

Definition 9.22 Fix a set A of propositional variables. For an arbitrary sentence $\chi^{=}(\overline{B},\overline{C}) \in \mathtt{BF}(A)$ we define

$$\big(\chi^{=}(\overline{B},\overline{C})\big)^* := \chi(\overline{B},\overline{C}),$$

and we extend this translation to the set $\mathtt{Dis}(\mathtt{BF}(A))$, simply by putting

$$\big(\bigvee_i \alpha_i\big)^* := \bigvee_i \alpha_i^*.$$

By Proposition 9.17 we may extend this definition to a map $(\cdot)^* : \mathtt{1FOE}(A) \to \mathtt{1FO}(A)$. ◁

Observe that the translation is in fact very simple: we obtain $\big(\chi^{=}(\overline{B},\overline{C})\big)^*$ from $\chi^{=}(\overline{B},\overline{C})$ simply by forgetting about the identity formulas occurring in the latter formula.

To exhibit the model-theoretic relation between the formulas α and α^*, we need one further definition.

Definition 9.23 Let $f : D' \twoheadrightarrow D$ be a surjective map from one set D' to another set D, and let A be some set of variables. Given a valuation $V : A \to \wp D$, we define the valuation $V_f : A \to \wp D'$, by putting, for $a \in A$:

$$V_f(a) := \{s' \in D' \mid f(s') \in V(a)\},$$

and, conversely, given a valuation $U : A \to \wp D'$, we let

$$U^f(a) := \{fs' \in D \mid s' \in U(a)\}$$

define a valuation on D. ◁

The only fact that we need about these translations and valuations is the following Proposition. We will use this result to transform the winning strategy of $\exists$ in one acceptance game to a winning strategy for her in a related acceptance game.

Proposition 9.24 *Let $\alpha \in \mathtt{1FOE}(A)$ be some one-step formula, and let D be some set. We let π denote the left projection map $\pi : D \times \omega \to D$.*
1) For any A-valuation V on D we have

$$D, V \models \alpha^* \text{ iff } D \times \omega, V_\pi \models \alpha. \tag{71}$$

2) As a corollary, for any A-valuation U on $D \times \omega$ we have

$$D \times \omega, U \models \alpha \text{ only if } D, U^\pi \models \alpha^*.$$

Proof. We leave the case where D is the empty set as an exercise for the reader, and focus on the case where $D \neq \varnothing$.

For part 1) of the Proposition, let α, D and π be as in its formulation. We will prove the equivalence (71).

For the left-to-right direction of (71), assume that $\langle D, V\rangle \models \chi(\overline{B}, \overline{C})$. Let $d_1, \ldots, d_n$ be elements in D satisfying the existential part of $\chi(\overline{B}, \overline{C})$, that is, for each i we find $d_i \in \bigcap_{b \in B_i} V(b)$. From the universal part of the formula it follows that for each $d \in D$ there is a subset $C_d \subseteq A$ such that $d \in \bigcap_{c \in C_d} V(c)$. Now we move to $D \times \omega$; it is easy to see that its elements $(d_1, 1), \ldots, (d_n, n)$ provide a sequence of n *distinct* elements that satisfy $(d_i, i) \in \bigcap_{b \in B_i} V_\pi(b)$ for each i. In addition, every element (d, n) distinct from the ones in the mentioned tuple will satisfy $(d, n) \in \bigcap_{c \in C_d} V_\pi(c)$. From these observations it is immediate that $\langle D \times \omega, V_\pi\rangle \models \chi^=(\overline{B}, \overline{C})$.

For the opposite direction of (71), assume that $\langle D \times \omega, V_\pi\rangle \models \chi^=(\overline{B}, \overline{C})$. Let $(d_1, k_1), \ldots, (d_n, k_n)$ be the sequence of distinct elements of $D \times \omega$ witnessing the existential part of $\chi^=(\overline{B}, \overline{C})$ in $\mathbb{D}'$. Then clearly, $d_1, \ldots, d_n$ witness the existential part of $\chi(\overline{B}, \overline{C})$ in $\langle D, V\rangle$. In order to show that $\langle D, V\rangle$ also satisfies the universal part $\forall z \bigvee_j \tau_{C_j}(z)$ of χ, consider an arbitrary element $d \in D$. Take any $m \in \omega \setminus \{k_1, \ldots, k_n\}$, then (d, m) is distinct from each (d_i, k_i). It follows that for some j we have $(d, m) \in \bigcap_{c \in C_j} V_\pi(c)$, and so we obtain $d \in \bigcap_{c \in C_j} V(c)$. Since d was arbitrary this shows that indeed $\langle D, V\rangle \models \forall z \bigvee_j \tau_{C_j}(z)$. So we have proved that $\langle D, V\rangle \models \chi(\overline{B}, \overline{C})$.

For part 2), assume that $D \times \omega, U \models \alpha^*$. It is straightforward to verify that $U(a) \subseteq (U^\pi)_\pi(a)$, for all $a \in A$. Hence by monotonicity of α with respect to the proposition letters in A, it follows that $D \times \omega, (U^\pi)_\pi \models \alpha^*$. But then we find $D, U^\pi \models \alpha^*$ by part 1) of the proposition. QED

Automata

Any translation between one-step formulas naturally induces a transformation of automata. In the current setting we obtain the following.

Definition 9.25 Given an automaton $\mathbb{A} = \langle A, \Theta, \Omega, a_I\rangle$ in $Aut(\mathtt{1FOE})$, we define the map $\Theta^* : (A \times \wp(\mathsf{P})) \to \mathtt{1FO}(A)$ by putting

$$\Theta^*(a) := (\Theta(a))^*,$$

and we let $\mathbb{A}^*$ denote the automaton $\mathbb{A}^* := \langle A, \Theta^*, \Omega, a_I\rangle$. $\lhd$

We have now arrived at the main technical result of this section. It involves the notion of the ω-unravelling $\mathbb{E}_\omega(\mathbb{S}, s)$ of a model $\mathbb{S}$ around a point s. This construction[5] generalizes that of the unravelling of a model (Definition 1.22).

Definition 9.26 Let κ be a countable cardinal with $1 \leq \kappa \leq \omega$, and let $(\mathbb{S}, s)$ be a pointed Kripke model of type (P, D). A *κ-path through* $\mathbb{S}$ is a finite (non-empty) sequence of the form

[5] In a later version of the notes, this construction will be defined in Chapter 1.

$s_0 d_1 k_1 s_1 \cdots s_{n-1} d_n k_n s_n$, where $s_i \in S$, $d_i \in \mathsf{D}$ and $k_i < \kappa$ for each i, and such that $R_{d_{i+1}} s_i s_{i+1}$ for each $i < n$. The set of such paths is denoted as $Paths^{\kappa}(\mathbb{S})$; we use the notation $Paths^{\kappa}_s(\mathbb{S})$ for the set of paths starting at s. Given such a sequence ρ, we let $last(\rho) \in S$ denote its last item.

The *κ-expansion* of $\mathbb{S}$ *around* s is the transition system $\mathbb{E}_{\kappa}(\mathbb{S}, s) = \langle Paths^{\kappa}_s(\mathbb{S}), \sigma^{\kappa} \rangle$, where

$$\begin{array}{rcl} \sigma^{\kappa}_V(s_0 \cdots d_n k_n s_n) & := & \sigma_V(s_n), \\ \sigma^{\kappa}_d(s_0 \cdots d_n k_n s_n) & := & \{(s_0 \cdots d_n k_n s_n dkt) \in Paths_s(\mathbb{S}) \mid R_d s_n t, 0 < k < \kappa\}. \end{array}$$

defines the coalgebra map $\sigma^{\kappa} = (\sigma_V, (\sigma_d \mid d \in \mathsf{D}))$. $\lhd$

It is not hard to check that the *unravelling* of a model (Definition 1.22) can be identified with its 1-expansion. It is straightforward to verify the following proposition.

Proposition 9.27 *For any countable cardinal κ with $1 \leq \kappa \leq \omega$, the function last, mapping a sequence to its last item, is a surjective bounded morphism from $\mathbb{E}_{\kappa}(\mathbb{S}, s)$ to $\mathbb{S}$ mapping the single-item sequence s to its single state s.*

Proposition 9.28 *Let $\mathbb{A}$ be an automaton in $Aut(\mathtt{1FOE})$, then for any pointed Kripke model $(\mathbb{S}, s)$ we have that*

$$\mathbb{S}, s \Vdash \mathbb{A}^* \text{ iff } \mathbb{E}_{\omega}(\mathbb{S}, s), s \Vdash \mathbb{A}. \tag{72}$$

Proof. Let $\mathbb{A} = \langle A, \Theta, \Omega, a_I \rangle$ and $(\mathbb{S}, s)$ be as in the formulation of the Theorem. Let f denote the (surjective) bounded morphism from $\mathbb{E}_{\omega}(\mathbb{S}, s)$ to $\mathbb{S}$, and recall that by definition f is the function *last* mapping an ω-path to its final element. We will only prove the right-to-left direction of (72), leaving the (slightly easier) opposite direction as an exercise to the reader.

So assume that $\mathbb{E}_{\omega}(\mathbb{S}, s), s \Vdash \mathbb{A}$. Then $\exists$ has a (positional) winning strategy h in the acceptance game $\mathcal{A}^{\omega} := \mathcal{A}(\mathbb{A}, \mathbb{E}_{\omega}(\mathbb{S}, s))@(a_0, s_0)$, where we write $a_0 := a_I$ and $s_0 := s$. We need to provide her with a winning strategy h' in the acceptance game $\mathcal{A} := \mathcal{A}(\mathbb{A}^*, \mathbb{S})@(a_0, s_0)$, and we will define h' by induction on the length of a partial $\mathcal{A}$-match $\Sigma = (a_i, s_i)_{0 \leq i \leq n}$. Via a simultaneous induction we define a partial $\mathcal{A}^{\omega}$-match $\Sigma' = (a_i, s'_i)_{0 \leq i \leq n}$ which will be guided by $\exists$'s winning strategy h and satisfies $f(s'_i) = s_i$, for all i.

For the inductive step of these definitions, consider a partial $\mathcal{A}$-match $\Sigma = (a_i, s_i)_{0 \leq i \leq n}$. Without loss of generality we may assume that Σ itself is guided by h', and inductively we may assume the existence of an h-guided shadow match $\Sigma' = (a_i, s'_i)_{0 \leq i \leq n}$ of $\mathcal{A}^{\omega}$ such that $f(s'_i) = s_i$, for all i. In order to extend the definition of h', so that it defines a move for $\exists$ in the partial match Σ, obviously we consider this partial shadow match. Let $U : A \to \wp\sigma^{\omega}_R(s')$ be the A-valuation picked by $\exists$'s winning strategy h in the match Σ'. If we compare the collections $\sigma_R(s)$ and $\sigma^{\omega}_R(s')$ of successors of s and s' respectively, it is obvious that f restricts to a surjection from $\sigma^{\omega}_R(s')$ to $\sigma_R(s)$. Hence we may take the valuation

$$U^f : A \to \wp\sigma_R(s),$$

induced by U as in Definition 9.23, as the move given by the strategy h in the partial match Σ.

To see that this move is legitimate, we need to show that

$$\sigma_R, U^f \models \Theta^*(a_n, \sigma_V(s_n)), \tag{73}$$

that is, the one-step formula $\Theta^*(a_n, \sigma_V(s_n)$ holds in the A-structure (σ_R, U^f). It will be convenient to think of $\sigma_R^\omega(s')$ as the set $\sigma_R(s) \times \omega$, and of f as the projection map $\pi : \sigma_R(s) \times \omega \to \sigma_R(s)$. Then (73) is immediate by Proposition 9.242) and the fact that

$$\sigma_R^\omega, U \models \Theta(a_n, \sigma_V(s_n)), \tag{74}$$

simply because the valuation U is the legitimate move provided by $\exists$'s winning strategy h. Clearly then, the valuation U^f is a legitimate move for $\exists$.

In order to finish the inductive definition, we need to show how to extend, for any response (b, t) of $\forall$ to $\exists$'s move U^f, the shadow match Σ' with a position (b, t') such that $ft' = t$. But this is straightforward: if (b, t) is a legitimate move for $\forall$ in $\mathcal{A}$ at position U, then we have $t \in U^f(b)$, and so by definition there is a state $t' \in \sigma_R^\omega(s')$ such that $ft' = t$ and $t' \in U(b)$. Clearly then the continuation $\Sigma' \cdot (b, t')$ of Σ' satisfies the requirements.

We will now show that the just defined strategy h' is in fact *winning* for $\exists$ in $\mathcal{A}$. For this purpose, consider a full $\mathcal{A}$-match Σ which is guided by h'.

First consider the case where Σ is finite. It is not hard to prove, using the existence of the h-guided shadow match Σ', that the player who got stuck in Σ is $\forall$.

Having taken care of the finite matches, we now consider the case where $\Sigma = (a_i, s_i)_{0 \leq i < \omega}$ is infinite. It is not difficult to see that in this case there is an h-guided infinite shadow match $\Sigma' = (a_i, s'_i)_{0 \leq i < \omega}$ of $\mathcal{A}^\omega$, such that $fs'_i = s_i$ for all $i < \omega$. But since h was assumed to be a winning strategy for $\exists$ in $\mathcal{A}^\omega$, Σ' is actually won by her. But since the priority maps of $\mathbb{A}$ and $\mathbb{A}^*$ are exactly the same, from this it is immediate that $\exists$ is also the winner the $\mathcal{A}$-match Σ.
QED

Proof of main result

As we shall see now, the expressive completeness of the modal μ-calculus is an almost immediate corollary of Proposition 9.28, given our earlier automata-theoretic characterizations of MSO and the modal μ-calculus.

Proof of Theorem 9.21. Let $\varphi \in$ MSO be a monadic second-order formula, and let $\mathbb{B}_\varphi \in$ *Aut*(1FOE) be the automaton as given in Theorem 9.9. Then by Theorem 9.8 there is a formula $\varphi^* \in \mu$ML that is equivalent to the translation $(\mathbb{B}_\varphi)^*$ of $\mathbb{B}_\varphi$. Clearly then φ^* has been effectively obtained from φ.

We will show that φ is invariant under bisimulations iff it is equivalent to the formula φ^*. The direction from right to left is immediate since formulas of the modal μ-calculus are bisimulation invariant.

For the opposite direction, observe that by Proposition 9.28 and the definition on φ^*, for an arbitrary pointed Kripke model $(\mathbb{S}, s)$ we have

$$\mathbb{S}, s \Vdash \varphi^* \text{ iff } \mathbb{E}_\omega(\mathbb{S}, s), s \Vdash \varphi. \tag{75}$$

Now assume that φ is bisimulation invariant, then we have that

$$\mathbb{S}, s \Vdash \varphi \text{ iff } \mathbb{E}_\omega(\mathbb{S}, s), s \Vdash \varphi. \tag{76}$$

Combining these two observations, we see that $\mathbb{S}, s \Vdash \varphi^*$ iff $\mathbb{S}, s \Vdash \varphi$. But since $(\mathbb{S}, s)$ was arbitrary, this means that φ and φ^* are equivalent, as required. QED

Notes

The result that the modal μ-calculus is the bisimulation-invariant fragment of monadic second-order logic is due to Janin & Walukiewicz [13].

Exercises

Exercise 9.1 Let (D, V) and (D', V') be two one-step models over the same set A of monadic predicates. Then (D, V) is a *quotient* of (D', V') if there is a surjection $f : D' \to D$ such that $V' = V_f$. An `MFOE`-sentence α is *invariant under taking quotients* if we we have that $(D, V) \models \alpha$ iff $(D', V') \models \alpha$, whenever (D, V) is a quotient of (D', V').

Let α be an `MFOE`-sentence. Prove that α is invariant under taking quotients iff $\alpha \equiv \alpha^*$. Conclude that `1FO` is the 'quotient-invariant fragment' of `1FOE`.

A Mathematical preliminaries

Sets 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 $\sim_S X := S \setminus X$. The size or cardinality of a set S is denoted as $|S|$.

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 functions 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, often called 'currying':

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

which associates, with a function $f : A \times B \to C$, the map that, for each $a \in A$, yields the function $f_a : B \to C$ given by $f_a(b) := f(a, b)$.

Relations Given a relation $R \subseteq A \times B$, we introduce the following notation. $\mathsf{Dom}(R)$ and $\mathsf{Ran}(R)$ denote the domain and range of R, respectively. R^{-1} denotes the converse of R. For $R \subseteq S \times S$, R^* denotes the reflexive-transitive closure of R, and R^+ the transitive closure. 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 $[R]Y = A \setminus \langle R\rangle(B \setminus Y)$. A relation R on S is *acyclic* if there are no elements s such that R^+ss.

An *equivalence relation* on a set A is a binary relation that is reflexive, symmetric and transitive. The *equivalence class* or *cell* of an element $a \in A$ relative to an equivalence relation is the set of all elements in A that are linked to a by the relation.

A *preorder* is a structure $(P, \sqsubseteq)$ such that $\sqsubseteq$ is a reflexive and transitive relation on P; given such a relation we will write $\sqsubset$ for the asymmetric version of $\sqsubseteq$ (given by $u \sqsubset v$ iff $u \sqsubseteq v$ but not $v \sqsubseteq u$) and $\equiv$ for the equivalence relation induced by $\sqsubseteq$ (given by $u \equiv v$ iff $u \sqsubseteq v$ and $v \sqsubseteq u$). Cells of such a relation will often be called *clusters*. A preorder is directed if for any two points u and v there is a w such that $u \sqsubseteq w$ and $v \sqsubseteq w$. A *partial order* is a preorder $\sqsubseteq$ which is antisymmetric, i.e., such that $p \sqsubseteq q$ and $q \sqsubseteq p$ imply $p = q$.

Sequences, lists and streams Given a set C, we define C^* as the set of finite *lists*, *words* or *sequences* over C. We will write ε for the empty sequence, and define $C^+ := C^* \setminus \{\varepsilon\}$ as the set of nonempty words. An infinite word, or *stream* over C is a map $\gamma : \omega \to C$ mapping natural numbers to elements of C; the set of these maps is denoted by C^ω. We write $\Sigma^\infty := \Sigma^* \cup \Sigma^\omega$ for the set of all sequences over Σ. The concatenation of a (finite) word u and a (finite or infinite) word v is denoted as $u \cdot v$ or uv.

We 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, *first*(π) denotes the first element of π. In the case that π is finite and nonempty we write *last*(π) for the last element of π. Given a stream $\gamma = c_0c_1\dots$ and two natural numbers $i < j$, we let $\gamma[i,j)$ denote the finite word $c_ic_{i+1}\dots c_{j-1}$.

Graphs and trees A *(directed) graph* is a pair $\mathbb{G} = \langle G, E\rangle$ consisting of a set G of *nodes* or *vertices* and a binary *edge* relation E on G. A *finite path* through such a graph is a sequence $(s_i)_{i\leq n} = s_0 \cdots s_n$ in G^* such that Es_is_{i+1} for all $i < n$. Similarly, an *infinite path* is a sequence $(s_i)_{0\leq i<\omega} = s_0s_1\cdots$ in G^ω such that Es_is_{i+1} for all $i < \omega$. A (proper) *cycle* is a path $s_0\cdots s_n$ such that $n > 0$, $s_0 = s_n$ and $s_0,\dots,s_{n-1}$ are all distinct. A graph is *acyclic* if it has no cycles. A *tree* is a graph $\mathbb{T} = (T, R)$ which contains a node r, called a *root* of $\mathbb{T}$, such that every element $t \in T$ is reachable by a *unique* path from r. (In particular, this means that $\mathbb{T}$ is acyclic, and that the root is unique.)

Fact A.1 (König's Lemma) *Let $\mathbb{G}$ be a finitely branching, acyclic tree. If $\mathbb{G}$ is infinite, then it has an infinite path.*

Order and lattices 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 $\bigwedge X$. Generally, given a statement S about ordered sets, we obtain its *dual statement* by replacing each occurrence of $\leq$ with $\geq$ and vice versa. The following principle often reduces our work load by half;

Order Duality Principle If a statement holds for all ordered sets, then so does its dual statement.

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 $\mathbb{C} = \langle C, \bigvee, \bigwedge\rangle$. Key examples of complete lattices are full power set algebras: given a set S, it is easy to show that the structure $\langle\wp(S), \bigcup, \bigcap\rangle$ is a complete lattice.

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 $i \in I$. 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 coordinatewise. 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

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

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

Ordinals A set S is *transitive* if $S \subseteq \wp(S)$; that is, if every element of S is a subset of S, or, equivalently, if $S'' \in S' \in S$ implies that $S'' \in S$. An *ordinal* is a transitive set of which all elements are also transitive. From this definition it immediately follows that any element of an ordinal is again an ordinal. We let $\mathcal{O}$ denote the class of all ordinals, and use lower case Greek symbols (α, β, γ, ..., λ, ...) to refer to individual ordinals.

The smallest, *finite*, ordinals are

$$\begin{array}{lll} 0 & := & \varnothing \\ 1 & := & \{0\} \quad (= \{\varnothing\}) \\ 2 & := & \{0,1\} \quad (= \{\varnothing, \{\varnothing\}\}) \\ 3 & := & \{0,1,2\} \quad (= \{\varnothing, \{\varnothing\}, \{\varnothing, \{\varnothing\}\}\}) \\ \vdots & & \end{array}$$

In general, the *successor* $\alpha + 1$ of an ordinal α is the set $\alpha \cup \{\alpha\}$; it is easy to check that $\alpha + 1$ is again an ordinal. Ordinals that are not the successor of an ordinal are called *limit* ordinals. Thus the smallest limit ordinal is 0; the next one is the first infinite ordinal

$$\omega := \{0, 1, 2, 3, \ldots\}.$$

But it does not stop here: the successor of ω is the ordinal $\omega+1$, etc. It is important to realize that there are in fact too many ordinals to form a set: $\mathcal{O}$ is a proper class. As a consequence, whenever we are dealing with a function $f : \mathcal{O} \to A$ from $\mathcal{O}$ into some set A, we can conclude that there exist distinct ordinals $\alpha \neq \beta$ with $f(\alpha) = f(\beta)$. (Such a function f will also be a class, not a set.)

We define an ordering relation $<$ on ordinals by:

$$\alpha < \beta \text{ if } \alpha \in \beta.$$

From this definition it follows that $\alpha = \{\beta \text{ in } \mathcal{O} \mid \beta < \alpha\}$ for every ordinal α. The relation $<$ is obviously transitive (if we permit ourselves to apply such notions to relations that are classes, not sets). It follows from the axioms of ZFC that $<$ is in fact *linear* (that is, for any two ordinals α and β, either $\alpha < \beta$, or $\alpha = \beta$, or $\beta < \alpha$) and *well-founded* (that is, every non-empty set of ordinals has a smallest element).

The fact that $<$ is well-founded allows us to generalize the principle of induction on the natural numbers to the transfinite case.

Transfinite Induction Principle In order to prove that all ordinals have a certain property, it suffices to show that the property is true of an arbitrary ordinal α whenever it is true of all ordinals $\beta < \alpha$.

A proof by transfinite induction typically contains two cases: one for successor ordinals and one for limit ordinals (the base case of the induction is then a special case of a limit ordinal). Analogous to the transfinite inductive proof principle there is a *Transfinite Recursion Principle* according to which we can construct an ordinal-indexed sequence of objects.

B Some remarks on computational complexity

▶ `Various general remarks to be supplied`

In computer science, $\mathcal{O}$ notation is used to classify algorithms according to their time and space complexity. Roughly, it is a way to state bounds that ignores multiplicatieve constants and low order terms.

Definition B.1 Let f and g be functions from the natural numbers to the natural numbers. We say that $f = \mathcal{O}(g)$ if there are positive constants c and k such that $f(n) \leq c \cdot g(n)$, for all $n \geq k$. $\lhd$

Quasi-polynomial time algorithms are algorithms that run longer than polynomial time, yet not so long as to be exponential time. The worst case running time of a quasi-polynomial time algorithm is $2^{\mathcal{O}((\log n)^c)}$ for some fixed $c > 0$. Note that for $c = 1$ we get a polynomial time algorithm.

References

[1] P. Aczel. An introduction to inductive definitions. In J. Barwise, editor, *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*, chapter C.5, pages 739–782. North-Holland Publishing Co., Amsterdam, 1977.

[2] A. Arnold and D. Niwiński. *Rudiments of μ-calculus*, volume 146 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 2001.

[3] J. van Benthem. *Modal Correspondence Theory*. PhD thesis, Mathematisch Instituut & Instituut voor Grondslagenonderzoek, University of Amsterdam, 1976.

[4] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Number 53 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.

[5] F. Bruse, O. Friedmann, and M. Lange. On guarded transformation in the modal μ-calculus. *Logic Journal of the IGPL*, 23(2):194–216, 2015.

[6] J.R. Büchi. On a decision method in restricted second order arithmetic. In E. Nagel, editor, *Proceedings of the International Congress on Logic, Methodology and the Philosophy of Science*, pages 1–11. Stanford University Press, 1962.

[7] A. Chagrov and M. Zakharyaschev. *Modal Logic*, volume 35 of *Oxford Logic Guides*. Oxford University Press, 1997.

[8] G. D'Agostino and M. Hollenberg. Logical questions concerning the μ-calculus. *Journal of Symbolic Logic*, 65:310–332, 2000.

[9] S. Demri, V. Goranko, and M. Lange. *Temporal Logics in Computer Science: Finite-State Systems*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2016.

[10] E.A. Emerson and C.S. Jutla. The complexity of tree automata and logics of programs (extended abstract). In *Proceedings of the 29th Symposium on the Foundations of Computer Science*, pages 328–337. IEEE Computer Society Press, 1988.

[11] E.A. Emerson and C.S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *Proceedings of the 32nd Symposium on the Foundations of Computer Science*, pages 368–377. IEEE Computer Society Press, 1991.

[12] D. Janin and I. Walukiewicz. Automata for the modal μ-calculus and related results. In *Proceedings of the Twentieth International Symposium on Mathematical Foundations of Computer Science, MFCS'95*, volume 969 of *LNCS*, pages 552–562. Springer, 1995.

[13] D. Janin and I. Walukiewicz. On the expressive completeness of the propositional μ-calculus w.r.t. monadic second-order logic. In *Proceedings of the Seventh International Conference on Concurrency Theory, CONCUR '96*, volume 1119 of *LNCS*, pages 263–277, 1996.

[14] B. Knaster. Un théorème sur les fonctions des ensembles. *Annales de la Societé Polonaise de Mathematique*, 6:133–134, 1928.

[15] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[16] D. Kozen. A finite model theorem for the propositional μ-calculus. *Studia Logica*, 47:233–241, 1988.

[17] D. Kozen and R. Parikh. A decision procedure for the propositional μ-calculus. In *Proceedings of the Workshop on Logics of Programs 1983*, LNCS, pages 313–325, 1983.

[18] R. McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and Control*, 9:521–530, 1966.

[19] L. Moss. Coalgebraic logic. *Annals of Pure and Applied Logic*, 96:277–317, 1999. (Erratum published *Ann.P.Appl.Log.* 99:241–259, 1999).

[20] A.W. Mostowski. Regular expressions for infinite trees and a standard form of automata. In A. Skowron, editor, *Computation Theory*, LNCS, pages 157–168. Springer-Verlag, 1984.

[21] D.E. Muller. Infinite sequences and finite machines. In *Proceedings of the 4th IEEE Symposium on Switching Circuit Theory and Logical Design*, pages 3–16, 1963.

[22] D. Niwiński. On fixed point clones. In L. Kott, editor, *Proceedings of the 13th International Colloquium on Automata, Languages and Programming (ICALP 13)*, volume 226 of *LNCS*, pages 464–473, 1986.

[23] D. Park. Concurrency and automata on infinite sequences. In *Proceedings 5th GI Conference*, pages 167–183. Springer, 1981.

[24] A. Pnueli. The temporal logic of programs. In *Proc. 18th Symp. Foundations of Computer Science*, pages 46–57, 1977.

[25] V.R. Pratt. Semantical considerations on Floyd-Hoare logic. In *Proc. 17th IEEE Symposium on Computer Science*, pages 109–121, 1976.

[26] S. Safra. On the complexity of ω-automata. In *Proceedings of the 29th Symposium on the Foundations of Computer Science*, pages 319–327. IEEE Computer Society Press, 1988.

[27] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[28] T. Wilke. Alternating tree automata, parity games, and modal μ-calculus. *Bulletin of the Belgian Mathematical Society*, 8:359–391, 2001.

[29] W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200:135–183, 1998.