Lectures on the modal μ-calculus

Yde Venema*

Abstract

These notes give an introduction to the theory of the modal μ-calculus.

*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*. In case D is a singleton, we will simply write R for the unique accessibility relation in a model.

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 $\mathbb{S}$. ◁

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 $\sigma(s) := (\sigma_V(s), \sigma_R(s))$. ◁

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*. $\lhd$

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*. 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. $\lhd$

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.

We will need some definitions and notations concerning atomic formulas.

Definition 1.6 Let P be a set of propositional variables. We define the sets $\mathtt{Lit}(\mathsf{P})$ and $\mathtt{At}(\mathsf{P})$ of, respectively, *literals* and *atomic formulas over* P as follows:

$$\begin{array}{lcl} \mathtt{Lit}(\mathsf{P}) & := & \{p, \overline{p} \mid p \in \mathsf{P}\} \\ \mathtt{At}(\mathsf{P}) & := & \{\bot, \top\} \cup \mathtt{Lit}(\mathsf{P}) \end{array}$$

We will generally use the symbol ℓ to denote an arbitrary literal. $\lhd$

Remark 1.7 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 $\leftrightarrow$. $\lhd$

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

Definition 1.8 We define the collection $Sf(\xi)$ of subformulas of a modal formula ξ by the following induction on the complexity of ξ:

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

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, $|\xi| := |Sf(\xi)|$. ◁

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 1.9 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 $\mathbb{S}$. ◁

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

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

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 1.11 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 1.12 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 1.13 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 1.14 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 $\langle 2^*, V, \mathit{Succ}_0, \mathit{Succ}_1 \rangle$. $\lhd$

Remark 1.15 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 1.16 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 σ, we always denote the *opponent* of σ by $\overline{\sigma}$. 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) 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

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

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 1.17 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 $Sf(\xi) \times S$ 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 σ in an initialized game is a method that σ uses to select his moves during the play. Such a strategy is *winning for* σ if every match of the game (starting at the given position) is won by σ, provided σ plays according to this strategy. A position (φ, s) is *winning* for σ if σ has a winning strategy for the game initialized in that position. (Note that this definition applies to *all* positions, not only to the ones owned by σ.) The set of winning positions in $\mathcal{E}(\xi, \mathbb{S})$ for σ is denoted as $\mathrm{Win}_\sigma(\mathcal{E}(\xi, \mathbb{S}))$.

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

▶ `Example to be added`

Theorem 1.18 (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 1.19 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 pair $(s, s') \in Z$.
(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: $\mathbb{S}, s \leftrightarrows_{\mathsf{P},\mathsf{D}} \mathbb{S}', s'$ 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 '$\leftrightarrows$' rather than '$\leftrightarrows_{\mathsf{P},\mathsf{D}}$'. ◁

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

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

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 1.20. 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 1.21 (Hennessy-Milner Property) *Let $\mathbb{S}$ and $\mathbb{S}'$ be two finitely branching transition systems of the same type. Then*

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

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

Proof. The direction from left to right follows from Theorem 1.20. 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 1.22 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 1.23 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 $\mathit{Paths}(\mathbb{S})$; we use the notation $\mathit{Paths}_s(\mathbb{S})$ 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 $\langle \mathit{Paths}_s(\mathbb{S}), \vec{\sigma} \rangle$, where the coalgebra map $\vec{\sigma} = (\vec{\sigma}_V, (\vec{\sigma}_d \mid d \in \mathsf{D}))$ is given by putting

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

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 1.20.

Theorem 1.24 (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 1.25 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 1.19 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 $(t, t') \in Z$. $\lhd$

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 1.26 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 σ if σ has a winning strategy for the game initialized in that position. The set of these positions is denoted as $\mathrm{Win}_\sigma(\mathcal{B}(\mathbb{S}, \mathbb{S}'))$. $\lhd$

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
$(s, s') \in S \times S'$	$\exists$	$\{Z \in \wp(S \times S') \mid Z$ is a local bisimulation for s 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 1.27 *Let* $(\mathbb{S}, s)$ *and* $(\mathbb{S}', s')$ *be two pointed Kripke models. Then* $\mathbb{S}, s \leftrightarrows \mathbb{S}', s'$ *iff* $(s, s') \in \mathrm{Win}_\exists(\mathcal{B}(\mathbb{S}, \mathbb{S}'))$.

Proof. For the direction from left to right: suppose that Z is a bisimulation between $\mathbb{S}$ and $\mathbb{S}'$ linking s and s'. Suppose that $\exists$, starting from position (s, s'), always chooses the relation Z itself as the local bisimulation. 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 1.28 ▶ `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 1.29 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 1.30 *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*

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

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 Complete derivation systems for modal logic

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. We first restrict attention to the monomodal case.

Definition 1.31 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 1.32 Observe that this definition involves the $\forall\exists\&\forall\exists$ pattern that we know from the definition of a bisimulation. The fundamental concept is 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. To be more precise, 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. $\lhd$

Remark 1.33 In the special case where $\Phi = \varnothing$ we find that $\mathbb{S}, s \Vdash \nabla\varnothing$ iff $R[s] = \varnothing$, that is, s has *no* successors. Using this it is easy to see that $\top = \nabla\{\top\} \vee \nabla\varnothing$. $\lhd$

Given that ∇ and $\{\Diamond, \Box\}$ are mutually expressible, we obtain an expressively equivalent language $\mathtt{ML}_\nabla$ if we replace $\Box$ and $\Diamond$ with the cover modality. As we will see further on it will be convenient for us to use a format for this language in which not only the cover modality, but also the disjunction and conjunction connectives take finite sets of formulas as their argument. That is, rather then working with disjunction and conjunction as *binary* connectives, we will work with their *finitary* versions. This perspective also allows us to omit the constants $\bot$ and $\top$ from the basic syntax, since we may consider them as abbreviations: $\bot := \bigvee\varnothing$ and $\top := \bigwedge\varnothing$.

Definition 1.34 The formulas of the language $\mathtt{ML}_\nabla$ are given by the following grammar:

$$\varphi \ ::= \ p \mid \overline{p} \mid \bigvee\Phi \mid \bigwedge\Phi \mid \nabla\Phi$$

where p is a propositional variable, and $\Phi \subseteq \mathtt{ML}_\nabla$. $\lhd$

Proposition 1.35 *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 1.29 that a relation $Z \subseteq A \times A'$ is *full on* A and A' if $(A, A') \in \overline{\wp}Z$, or in other words: $A \subseteq \mathsf{Dom}(Z)$ and $A' \subseteq \mathsf{Ran}(Z)$.

Proposition 1.36 (Binary Modal Distributive Law) *Let* Φ *and* Φ' *be two sets of formulas. Then the following two formulas are equivalent:*

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

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 1.36 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 pair consisting of a set of literals and a ∇-formula (or, a certain set of ∇-formulas in the polymodal case). The intended reading of the bullet operator is as follows:

$$\alpha \bullet \Phi \equiv (\bigwedge \alpha) \wedge \nabla\Phi.$$

Definition 1.37 Fix a finite set P of proposition letters. Then the set $\mathtt{DML}(\mathsf{P})$ of *disjunctive monomodal formulas* in P is given by the following grammar:

$$\varphi \ ::= \ \top \ \mid \ \bigvee\Phi \ \mid \ \alpha \bullet \Phi,$$

where α is a finite set of literals over P and Φ is a finite set of formulas in $\mathtt{DML}(\mathsf{P})$. $\lhd$

Note that the proposition letters in P and their negations themselves do not qualify as disjunctive formulas. However, these formulas are easily seen to be equivalent to disjunctive formulas: for instance, we have $\ell \equiv \{\ell\} \bullet \{\top\} \vee \{\ell\} \bullet \varnothing$, for any literal ℓ.

Remark 1.38 In the above definition we do not need to list the formula $\bot$ explicitly as a disjunctive formula, since we can see it as an abbreviation: $\bot := \bigvee \varnothing$. This is different for the formula $\top$, however. Since we no longer have $\bigwedge$ as a connective, we cannot use it to define $\top$. For this reason we have added $\top$ as a primitive constant. $\lhd$

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

Theorem 1.39 *Let P be a set of proposition letters. Then there are effective ways to transform an arbitrary formula in $\mathtt{ML}(\mathsf{P})$ into an equivalent formula in $\mathtt{DML}(\mathsf{P})$, and vice versa. As a corollary, the languages $\mathtt{ML}(\mathsf{P})$ and $\mathtt{DML}(\mathsf{P})$ are expressively equivalent.*

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

Remark 1.40 In the polymodal case we adapt the definition as follows. Let $\Phi = \{\Phi_d \mid d \in \mathsf{D}\}$ be a D-indexed family of formula sets. Then we write $\nabla_{\mathsf{D}}\Phi := \bigwedge_{d\in\mathsf{D}} \nabla_d\Phi_d$, where ∇_d is the cover modality associated with the action d. The following grammar defines the set $\mathtt{DML}_{\mathsf{D}}(\mathsf{P})$ of *disjunctive polymodal formulas* in D and P

$$\varphi \ ::= \ \top \ \mid \ \bigvee\Phi \ \mid \ \alpha \bullet \Phi,$$

where $\alpha \subseteq_\omega \mathtt{Lit}(\mathsf{P})$ and Φ is an D-indexed family of finite sets of $\mathtt{DML}_{\mathsf{D}}(\mathsf{P})$-formulas. One may then formulate and prove a polymodal version of Theorem 1.39, relating $\mathtt{ML}_{\mathsf{D}}$ and $\mathtt{DML}_{\mathsf{D}}$. $\lhd$

Notes

Modal logic has a long history in philosophy and mathematics, for an overview we refer to Blackburn, de Rijke and Venema [3]. The use of modal formalisms as specification languages in process theory goes back at least to the 1970s, with Pratt [19] and Pnueli [18] 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 [2], 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 [17]. 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 [13] and Janin & Walukiewicz [7].

Readers who want to study modal logic in more detail are referred to Blackburn, de Rijke and Venema [3] or Chagrov & Zakharyaschev [5].

Exercises

Exercise 1.1 Prove Theorem 1.18.

Exercise 1.2 Prove that the Hennessy-Milner theorem (Theorem 1.21) 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 $\mathbb{S}, s \leftrightarrows_\alpha \mathbb{S}', s'$ 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 $\mathbb{S}, s \leftrightarrows_\omega \mathbb{S}', s'$ iff $\mathbb{S}, s \leftrightarrows_k \mathbb{S}', s'$ for all $k < \omega$.

(a) Give concrete examples such that $\mathbb{S}, s \leftrightarrows_\omega \mathbb{S}', s'$ but not $\mathbb{S}, s \leftrightarrows \mathbb{S}', s'$.
(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 $\mathbb{S}', s'$:

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

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) $\mathbb{S}, s \leftrightarrows_\omega \mathbb{S}', s'$

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

(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 1.39.

2 The modal μ-calculus: basics

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, in two variants. 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, and we discuss some further syntactic issues.

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. $\lhd$

The modal μ-calculus allows one to explicitly refer to such smallest and largest solutions. For instance, 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. 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

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

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 interpret the formula $\varphi(x)$ as an *operation* $\varphi_x^{\mathbb{S}}$ 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. 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 the subformula-based 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 δ (in the example, the formula $p \vee \Diamond x$), so that the next position in the game is the pair (δ, s). The alternative version of the evaluation game also crucially involves unfolding: here, for instance, a position of the form $(\mu x\, \varphi, s)$ is replaced by the pair $(\varphi[\mu x\, \varphi/x], s)$ we obtain by unfolding the fixpoint formula $\mu x\, \varphi$.

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 or fixpoint formulas 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 Basic syntax

Formulas

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 $\overline{\nu} := \mu$. $\lhd$

A formula of the form $\eta x\, \varphi$ is called a *fixpoint formula*, and, more specifically, a *μ-formula* if $\eta = \mu$ and a *ν-formula* if $\eta = \nu$. Furthermore, conjunctions and disjunctions will sometimes be called *boolean* μML-formulas, and formulas of the form $\Diamond_d \varphi$ or $\Box_d$ will sometimes be called *modal*.

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 $\rightarrow$, 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 $\mu x \Big(\big((\overline{p} \vee \Box x) \vee y \big) \vee \nu y\, (q \wedge \Box(x \vee y)) \Big)$.

Remark 2.4 An alternative definition of the language of the modal μ-calculus makes a distinction between propositional *variables* and *proposition letters*. Formulas are now defined as follows:

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

where p is a proposition letter, x a propositional variable, and d is an atomic action. In this framework, only propositional variables can be bound. $\lhd$

Length and syntax tree of a formula

There are various ways to *measure* a μ-calculus formula. The most basic measure of a formula is its *length*, which basically corresponds to its number of symbols.

Definition 2.5 Given a μ-calculus formula ξ, we define its *length* $|\xi|^\ell$ inductively as follows:

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

$\lhd$

We assume that the reader is familiar with the concept of the *syntax tree* or *construction tree* $\mathbb{T}_\xi$ of a formula ξ. We will not give a formal definition of this structure, but confine ourselves to an example: in Figure 2.1 we display the syntax tree of the μ-calculus formula $\mu x.(\overline{p} \vee \Diamond x) \vee \nu y.(q \wedge \Box(x \vee y))$. Note that the *length* of a formula corresponds to the number of nodes of its syntax tree, and that an *occurrence* of a certain symbol in a formula may be associated with some node in the formula's syntax tree that is labelled with that symbol; occurrences of literals correspond to *leaves* of the tree.

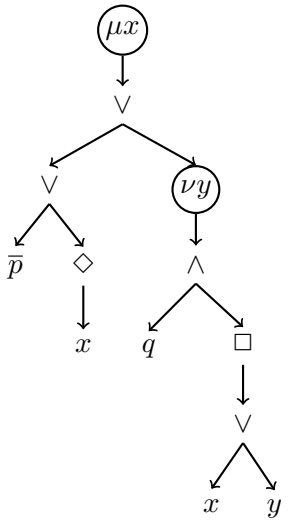

Figure 1: A syntax tree

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 2.6 We define the set $Sf_0(\xi)$ of *direct subformulas* of a formula $\xi \in \mu\mathtt{ML}$ via the following case distinction:

$$\begin{array}{lcll} Sf_0(\xi) & := & \varnothing & \text{if } \xi \in \mathtt{At}(\mathsf{P}) \\ Sf_0(\xi_0 \circledast \xi_1) & := & \{\xi_0, \xi_1\} & \text{where } \circledast \in \{\wedge, \vee\} \\ Sf_0(\heartsuit \xi_0) & := & \{\xi_0\} & \text{where } \heartsuit \in \{\Diamond, \Box\} \\ Sf_0(\eta x.\xi_0) & := & \{\xi_0\} & \text{where } \eta \in \{\mu, \nu\}, \end{array}$$

and we write $\varphi \lhd_0 \xi$ if $\varphi \in Sf_0(\xi)$.

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

The *(subformula) dag* of a formula ξ is defined as the directed acyclic graph $(Sf(\xi), \rhd_0)$, where $\rhd_0$ is the converse of the direct subformula relation $\lhd_0$. $\lhd$

▶ `Give an example comparing the syntax tree of a formula to its subformula dag.`

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

Definition 2.7 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:

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

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. $\lhd$

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 2.8 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 2.9 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 $p \vee \Diamond x$.

Remark 2.10 In the alternative definition of the language of the modal μ-calculus as discussed in Remark 2.4, just like in first-order logic one makes a difference between *(open) formulas* (which may contain free variables) and *sentences* (which may not). Observe that the sentences correspond to the tidy formulas in our framework. For instance, $\mu x\,(p \vee \Diamond x)$ is a sentence, $\mu x\,(y \vee \Diamond x)$ is an open formula, and $\mu p\,(x \vee \Diamond p)$ is not a well-formed formula (assuming that p is a proposition letter, and x is a variable). $\lhd$

Substitution

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 precise definition of a substitution operation needs some care. In particular, we need to protect the substitution operation from variable capture.

Example 2.11 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. $\lhd$

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

Definition 2.12 Let ξ and x be respectively a modal μ-calculus formula and a propositional variable. We define what it means for ξ to be *free for x in a formula φ* by the following induction on the complexity of φ:

- if φ is an atomic formula then ξ is free for x in φ, unless $\varphi = \overline{x}$[1];
- ξ is free for x in $\varphi_0 \circledast \varphi_1$ if it is free for x in both φ_0 and φ_1;
- ξ is free for x in $\heartsuit\psi$ if it is free for x in ψ;
- ξ is free for x in $\eta y\, \psi$ if $x \notin FV(\eta y\, \psi)$ or if $y \notin FV(\xi)$ and ξ is free for x in ψ. $\lhd$

Informally, ξ is *free for x in φ* if φ is positive in x and no free variable in ξ gets bound, after substitution, by a fixpoint operator in φ. A special case of this, that we shall encounter frequently, is the following.

Proposition 2.13 *Let φ, ξ and x be respectively two modal μ-calculus formulas and a propositional variable, such that $FV(\xi) \cap BV(\varphi) \subseteq \{x\}$. Then ξ is free for x in φ.*

Definition 2.14 Let $\{\xi_z \mid z \in Z\}$ 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 ξ_z is free for z in φ. We inductively define the *simultaneous substitution* $[\xi_z/z \mid z \in Z]$ as the following operation on $\mu\mathtt{ML}$:

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

In case Z is a singleton, say $Z = \{z\}$, we will simply write $\varphi[\xi_z/z]$, or $\varphi(\xi)$ if z is understood. $\lhd$

[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.

▶ Add some examples

Remark 2.15 In case ψ is not free for some $z \in Z$ in ξ, we take a standard approach using *alphabetical variants*. Roughly, two formulas are alphabetical variants if we can obtain one from the other by renaming bound variables. We then define a correct version of the substitution $\xi[\psi_z/z \mid z \in Z]$ as follows: first we take some canonically chosen alphabetical variant ξ' of ξ such that each ψ_z is free for z in ξ', and then we set

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

However, in almost all situations that we will encounter we will only need perform substitutions that are 'safe' in the sense that the substituted formula is free for the variable it replaces. This means that generally we may avoid taking alphabetical variants. Situations where this is not the case will be explicitly marked. The reason for taking such care is that the operation of taking alphabetical variants is not completely harmless when it comes to size issues. We will come back to this matter in more detail later. ◁ •

Unfolding

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 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 $\mu\mathtt{ML}$:

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

In words: every formula is equivalent to its *unfolding*.

Definition 2.16 Given a formula $\eta x.\chi \in \mu\mathtt{ML}$, we call the formula $\mathtt{unf}(\xi) := \chi[\eta x.\chi/x]$ its *unfolding*. ◁

Remark 2.17 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 (unless we take alphabetical variants). 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 $\overline{p} \wedge \nu p.\Box(x \vee p)$, naively unfolding the (untidy) formula $\mu x.\chi$ would produce the ungrammatical $\overline{p} \wedge \nu p.\Box\big((\mu x.\overline{p} \wedge \nu p.\Box(x \vee p)) \vee p\big)$. ◁

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 2.18 *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:

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

which can easily be proved. QED

Guardedness

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

▶ `A bit more to be said here.`

Definition 2.19 A variable x is *guarded* in a $\mu\mathtt{ML}_\mathsf{D}$-formula φ if every free 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. ◁

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

2.2 The evaluation game based on subformulas

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 2.20 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 $(\varphi, s) \in Sf(\xi) \times S$. 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). ◁

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). (In section 2.5 we present an alternative version in which the formula $\eta x\,\delta$ is replaced with its unfolding). 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 2.21 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 π, we will declare $\exists$ to be the winner of π if this variable is a ν-variable, and $\forall$ in case we are dealing with a μ-variable. But what happens in case that *various* variables are unfolded infinitely often?

Definition 2.22 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

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

(where κ is either a natural number or ω) which is in accordance with the rules of the evaluation game — that is, π 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 π, we let $Unf^\infty(\pi) \subseteq BV(\xi)$ denote the set of variables that are unfolded infinitely often during π. $\lhd$

As we will see now, for any infinite match π of the evaluation game, the set $Unf^\infty(\pi)$ contains a unique variable that ranks higher than all the others in the following order.

Definition 2.23 Given a clean formula ξ, we define the *unfolding order* $\trianglelefteq_\xi$ on the set $BV(\xi)$ by putting $x \trianglelefteq_\xi y$ if $\delta_x \trianglelefteq \delta_y$. If $x \trianglelefteq_\xi y$ we say that y *ranks higher* than x. $\lhd$

Proposition 2.24 *Let ξ be a clean $\mu\mathtt{ML}_\mathsf{D}$-formula, and $\mathbb{S}$ a labelled transition system. Then for any infinite match π of the game $\mathcal{E}(\xi, \mathbb{S})$, the set $Unf^\infty(\pi)$ has a highest ranking member, in terms of the unfolding order $\trianglelefteq_\xi$.*

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

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

To prove this, note that since y is $\trianglelefteq_{\xi}$-maximal in U, there must be a position in π such that y is unfolded to δ_y, while no higher ranking variable z is unfolded at any later position in π.

But then a straightforward induction shows that all formulas featuring at later positions must be subformulas of δ_y: the key case here is where $z \trianglelefteq \delta_y$ unfolds to δ_z. Here we distinguish two cases whether $z \in FV(\delta_y)$ or not; in the first case we use the claim below.

Claim 1 Let $\varphi \trianglelefteq \xi$ be such that $z \in FV(\varphi) \cap BV(\xi)$. Then $\varphi \trianglelefteq \delta_z$.

Proof of Claim Let ℓ_{φ} be the length of the shortest $\lhd_0$-path from φ to ξ. Then we prove by induction on ℓ_{φ} that $z \in FV(\varphi)$ implies $\varphi \trianglelefteq \delta_z$. The case where $\ell_{\varphi} = 0$ holds vacuously since here we have $\varphi = \xi$, and z is not free in ξ. In the inductive case we assume that $\ell_{\varphi} > 0$, and we consider a formula $\psi \trianglelefteq \xi$ of which φ is a direct subformula. We distinguish cases. If ψ is not of the form $\psi = \lambda z.\varphi$ then we have $z \in FV(\psi)$ so that by the inductive hypothesis we have $\varphi \trianglelefteq \psi \trianglelefteq \delta_z$. On the other hand, by cleanness of ξ, the formula ψ can only be of the form $\psi = \lambda z.\varphi$ if $\lambda = \eta_z$ and $\varphi = \delta_z$, so in this case we also have $\varphi \trianglelefteq \delta_z$. ◀

Returning to the main argument, if $z \in FV(\delta_y)$ then we may infer from Claim 1 that $\delta_y \trianglelefteq \delta_z$; this means that $y \trianglelefteq_{\xi} z$, so that by the assumption of maximality we must have $y = z$, so that δ_z is a subformula of δ_y indeed. On the other hand, if $z \notin FV(\delta_y)$ then it is not hard to see that the cleannes of ξ implies that $\delta_z \trianglelefteq \delta_y$. QED

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

Definition 2.25 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. ◁

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

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

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

Definition 2.26 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 $(\xi, s) \in \mathrm{Win}_{\exists}(\mathcal{E}(\xi, \mathbb{S}))$. ◁

Remark 2.27 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 2.28 Observe that we have defined the game-theoretic semantics for *clean* formula only. The reason for this should be obvious: at any position (φ, s) in the evaluation game it should be unequivocally clear what the admissible moves are, and in the case where φ is a (positive) literal this is only guaranteed if the ambient formula is clean. Consider for example the formula $\xi = \Diamond p \wedge (\mu p.q \vee \Diamond p) \wedge \nu p.\Box\Diamond p$, which has three occurrences of each of the subformulas $\Diamond p$ and p. It should be obvious that we cannot define an adequate evaluation game for this formula based on the set $Sf(\xi) \times S$ as positions (where S is the set of points in the Kripke model under consideration), since there is no reasonable definition of a legitimate move at a position of the form (p, s).

In the next section we define an alternative version of the evaluation game which works for the wider class of *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 2.29 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 2.30 Now we turn to the formulas $\mu x.\Diamond x$ and $\nu x.\Diamond x$. First consider how a match for any of these formulas proceeds. The first two positions of such a match will be of the form $(\eta x.\Diamond x, s)(\Diamond x, s)$, at which point it is $\exists$'s turn to make a move. Now she either is stuck (in case the state s has no successor) or else the next two positions are $(x, t)(\Diamond x, t)$ for some successor t of s, chosen by $\exists$. Continuing this analysis, we see that there are two possibilities for a match of the game $\mathcal{E}(\eta x.\Diamond x, \mathbb{S})$:

1. the match is an infinite sequence of positions

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

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

2. the match is 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 $s_0 R s_1 R \dots s_k$ through $\mathbb{S}$, where s_k has no successors.

Note too that in either case it is only $\exists$ who has turns, and that her strategy corresponds to choosing a *path* through $\mathbb{S}$. From this it is easy to derive that
• $\mu x.\Diamond x$ 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 2.31 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 $\mathbb{S}$:

$$\mathbb{S}, s \Vdash_g \xi \text{ iff there is some path from } s \text{ on which } p \text{ is true infinitely often.} \tag{9}$$

To see this, first suppose that there is a path $\pi = s_0 s_1 s_2 \ldots$ as described in the right hand side of (9) 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 π. 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 π 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 Bisimulation invariance and the 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 2.32 (Bisimulation Invariance) *Let ξ be a modal fixpoint formula with $FV(\xi) \subseteq \mathsf{P}$, and let $\mathbb{S}$ and $\mathbb{S}'$ be two labelled transition systems with points s and s', respectively. If $\mathbb{S}, s \leftrightarrows_{\mathsf{P}} \mathbb{S}', s'$, then*

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

Proof. Assume that $s \leftrightarrows_{\mathsf{P}} s'$ 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 π ending in a position for $\exists$, we let $f(\pi)$ 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 $s \leftrightarrows s'$ instead of $s \leftrightarrows_{\mathsf{P}} s'$).

- 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 $s \leftrightarrows s'$, there is a successor t' of s' such that $t \leftrightarrows t'$. This t' is $\exists$'s move in $\mathcal{E}'$, so that (ψ, t) and (ψ, t') are the next positions in $\mathcal{E}$ and $\mathcal{E}'$, respectively.

- 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 $t \leftrightarrows t'$. So we may assume that $\forall$ moves the game token of $\mathcal{E}$ to position (ψ, t).

- Finally, if $\xi = \eta x\, \delta_x$ then the next positions in $\mathcal{E}$ and $\mathcal{E}'$ are, respectively, (δ_x, s) and (δ_x, s').

The crucial observation is that if $\exists$ does not win immediately, then at least she can guarantee that the next positions in $\mathcal{E}$ and $\mathcal{E}'$ are of the form (φ, u) and (φ, u') respectively, with $u \leftrightarrows u'$, and such that the move in $\mathcal{E}$ is consistent with f. 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 $\pi = p_0 p_1 \dots p_n$ 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 $\pi' = (\varphi'_0, s'_0)(\varphi'_1, s'_1) \dots (\varphi'_n, s'_n)$ is *linked* to an

$\mathcal{E}$-match $\pi = (\varphi_0, s_0)(\varphi_1, s_1)\ldots(\varphi_n, s_n)$ (of the same length), if $\varphi'_i = \varphi_i$ and $\mathbb{S}', s'_i \leftrightarrows \mathbb{S}, s_i$ for all i with $0 \leq i \leq n$. The key claim in the proof states that, for a $\mathcal{E}'$-match π', 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 π' be a finite $\mathcal{E}'$-match, and assume that π' is linked to some f-guided $\mathcal{E}$-match π. Then one of the following two cases apply.

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

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

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 π, an f-guided $\mathcal{E}$-match which is linked to π'. For the base case of this induction, simply observe that by the assumption that $\mathbb{S}, s \leftrightarrows \mathbb{S}', s'$, 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 π', and inductively assume that there is a bisimilarity-linked f-guided $\mathcal{E}$-match π. Now distinguish cases:

- If $\mathsf{last}(\pi')$ is a position for $\exists$, we use item 1) of Claim 1 to define her move (φ, t'); it follows that $\pi' \cdot (\varphi, t')$ and $\pi \cdot (\varphi, t)$ are bisimilarity-linked (where (φ, t) is the move dicated by f in π).
- On the other hand, in case $\mathsf{last}(\pi')$ is a position for $\forall$, assume that he makes some move, say, (ψ, t'); now we use item 2) of the claim to define a continuation $\pi \cdot (\psi, t)$ of π that is bisimilarity-linked to $\pi' \cdot (\psi, t')$.

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

If π' is infinite, say $\pi' = (\varphi_i, s'_i)_{i<\omega}$, 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 $\pi = (\varphi_i, s_i)_{i<\omega}$ such that $\mathbb{S}', s'_i \leftrightarrows \mathbb{S}, s_i$ for all $i < \omega$. The key observation, however, is that the two sequences of formulas, in the $\mathcal{E}'$-match π' and its $\mathcal{E}$-shadow π, respectively, are exactly the *same*. This means that also in the infinite case the winner of π' is the winner of π, and since π 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 2.33 (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 1.23. 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 2.34 (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 $\pi = (\xi, r)(\varphi_1, t_1)\ldots$ which is consistent with f'. The key observation of the proof is that π is also a match of $\mathcal{E}@(\xi, r)$, that is consistent with f. To see this, simply observe that all moves of $\forall$ in π could have been made in the game on $\mathbb{T}$ as well, whereas by construction, all f' moves of $\exists$ in $\mathcal{E}'$ are f moves in $\mathcal{E}$.

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

2.5 The evaluation game based on the closure set

In this section we define an alternative version of the evaluation game for μ-calculus formulas, in which the equivalence

$$\eta x\, \chi \equiv \chi[\eta x\, \chi/x]$$

is exploited more directly than in the *subformula game* that we defined in section 2.2. The idea in the *closure game* is that, at a position $(\eta x\, \chi, s)$ the fixpoint formula will simply be unfolded, yielding the pair $(\chi[\eta x\, \chi/x], s)$ as the (unique) next position. That is, the admissible moves in the closure game are 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: Positions and admissible moves in the closure evaluation game $\mathcal{E}^c(\xi, \mathbb{S})$

In order to turn this table into a proper game, we need to introduce appropriate *winning conditions* for the two players. For this purpose we introduce some terminology and notation, and we make some observations. We start with the notion of a *trace*.

Traces and the closure game

Definition 2.35 Let $\to_C$ be the binary relation between tidy μ-calculus formulas given by the following exhaustive list:

1) $(\varphi_0 \circledast \varphi_1) \to_C \varphi_i$, for any $\varphi_0, \varphi_1 \in \mu\mathtt{ML}$, $\circledast \in \{\wedge, \vee\}$ 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 $i < \omega$. $\lhd$

Intuitively a trace is a sequence that corresponds to the formula part of a possible match of the closure game. The *closure* of a formula consists of the formulas that can be encountered in such a match.

Definition 2.36 We define the relation $\twoheadrightarrow_C$ as the reflexive and transitive closure of $\to_C$, and define the *closure* of a tidy formula ξ as the set

$$Cl(\xi) := \{\varphi \mid \xi \twoheadrightarrow_C \varphi\}.$$

Given a set of formulas Ψ, we put $Cl(\Psi) := \bigcup_{\xi\in\Psi} Cl(\xi)$, and we call Ψ *closed* if $\Psi = Cl(\Psi)$. Formulas in the set $Cl(\xi)$ are said to be *derived* from ξ. The *closure graph* of ξ is the directed graph $\mathbb{C}_\xi := (Cl(\xi), \to_C)$. $\lhd$

In words, $Cl(\xi)$ is the smallest set which contains ξ and is closed under direct boolean and modal subformulas, and under unfoldings of fixpoint formulas. In terms of traces: a formula χ belongs to the closure of a formula ξ iff there is a trace from ξ to χ. Furthermore, a trace starting at ξ is nothing but a path in the closure graph starting at ξ.

Remark 2.37 The final example of Remark 2.17 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 2.38 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 $Cl(\xi_1) = \Phi$; in fact, we have $Cl(\varphi) = \Phi$ for every formula $\varphi \in \Phi$. In Figure 2 we depict the *closure graph* of ξ_1.

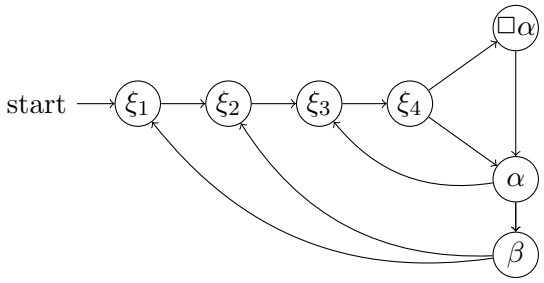

Figure 2: A closure graph

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, and that it is a subformula of *every* formula in $Cl(\xi_1)$. $\lhd$

The closure of ξ consists of the formulas that one may encounter in a match of the closure game $\mathcal{E}^c(\xi, \mathbb{S})$, and, as a consequence of this, we will take $Cl(\xi) \times S$ as the set of positions in this game. As we will see now, the key observation for defining the winning conditions of this game is that every infinite trace can be identified as either a μ-trace or a ν-trace. This is in some sense the analogon of Proposition 2.24.

Proposition 2.39 *1) Let τ be a finite trace. Then there is a unique formula on τ which is a subformula of every formula on τ.*

2) Let τ be a infinite trace. Then there is a unique formula which appears infinitely often on τ, and is a subformula of cofinitely many formulas on τ. This formula is always a fixpoint formula.

Proof. The reader is asked to supply a proof of this Proposition in Exercise 2.9. QED

Definition 2.40 Let τ be an infinite trace. The formula $\eta x\, \varphi$ which appear infinitely often on τ and is a subformula of all formulas on τ is called the *most significant formula* of τ, notation: $\mathtt{msf}(\tau)$. Depending on the nature of η we call τ either a *μ-trace* or a *ν-trace*.

For future reference we define some relations. We write $\rho \twoheadrightarrow_C^{\psi} \sigma$ if there is a trace $\rho = \chi_0 \to_C \chi_1 \to_C \cdots \to_C \chi_n = \sigma$ such that $\psi \trianglelefteq \chi_i$, for every $i \in [0, n]$. We write $\rho \twoheadrightarrow_C^{\eta} \sigma$ (for $\eta \in \{\mu, \nu\}$) if we have $\rho \twoheadrightarrow_C^{\psi} \sigma$ for some η-formula ψ, and $\rho \twoheadrightarrow_C^{o} \sigma$ if $\rho \twoheadrightarrow_C^{\psi} \sigma$ for some formula ψ which is not a fixpoint formula. $\lhd$

The classification of infinite traces as either μ- or ν-traces enables us to complete the definition of the closure game.

Definition 2.41 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 $Cl(\xi) \times S$, and the game graph (i.e., the partitioning of $Cl(\xi) \times S$ into positions for the two players, together with the set $E(z)$ of admissible moves at each position), is given in Table 5.

The winner of an infinite match $\pi = (\xi_n, s_n)_{n<\omega}$ is $\exists$ if its left projection $\pi_L := (\xi_n)_{n<\omega}$ is a ν-trace, and $\forall$ if it is a μ-trace. $\lhd$

The closure operation

The closure operation is one of the most fundamental tools in the theory of the modal μ-calculus, and in this subsection we discuss some of its properties, the most important being Proposition 2.45 stating that the closure of a finite set is always finite.

We first gather some basic observations. To start with, while Example 2.38 clearly shows that, while the unfolding of a clean formula will generally not be clean, tidyness is preserved.

Proposition 2.42 *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 and $\xi[\psi/x]$ is tidy then ψ is also free for x in φ, and $\varphi[\psi/x]$ is tidy.

Proof. The proofs of the first two 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 $\chi_0 \circledast \chi_1$ is so (where $\circledast \in \{\wedge, \vee\}$), 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 2.18.

For part 3) we also reason by induction on the length of the trace $\tau : \xi \twoheadrightarrow_C \varphi$. We focus on the key inductive case where $\tau : \xi \to_C \xi' \to_C \varphi$, and ξ' is a fixpoint formula, say, $\xi' = \eta y\, \chi$; in this case we find $\varphi = \chi[\xi'/y]$. The inductive hypothesis states that ψ is free for x in ξ' and that $\xi'[\psi/x]$ is tidy. By definition of freeness we can make the following case distinction. If x is not free in ξ' then it is not free in $\chi[\xi'/x]$ either, which immediate implies that ψ is free for x in $\chi[\xi'/x]$, and that $\varphi[\psi/x] = \chi[\xi'/x][\psi/x] = \chi[\xi'/x]$, so that $\varphi[\psi/x]$ is tidy by part 2).

In the remainder we focus on the case where $y \notin FV(\psi)$ and ψ is free for x in χ. Clearly it suffices to show that

$$\psi \text{ is free for } x \text{ in } \rho[\xi'/y], \text{ for every } \rho \trianglelefteq \chi. \tag{10}$$

We prove (10) by induction on ρ and only consider the key case, where ρ is a fixpoint formula, say, $\rho = \lambda z\, \rho'$. Then by the inner induction hypothesis we have that ψ is free for x in ρ'. Now assume for contradiction that ψ is *not* free for x in ρ'; this can only be the case if z itself is free in ψ. We leave it for the reader to verify that this implies that z has both free and bound occurrences in $\xi'[\psi/x]$, contradicting the tidyness of ξ'. QED

Second, the following proposition states that Cl is indeed a closure operation. We leave the proof of this proposition as an exercise for the reader.

Proposition 2.43 *Cl is a closure operation on the collection of tidy formulas:*

1) $\Phi \subseteq Cl(\Phi)$;
2) Cl is monotone: $\Phi \subseteq \Psi$ implies $Cl(\Phi) \subseteq Cl(\Psi)$;
3) $Cl(Cl(\Phi)) \subseteq Cl(\Phi)$.

The proposition below will prove to be very useful. It details how the closure map interacts with various connectives and formula constructors of the μ-calculus.

Proposition 2.44 *Let ξ be a tidy formula. Then the following hold.*

1) Let $\ell \trianglelefteq \xi$ be a literal occurring in ξ, and assume that $\ell \notin BV(\xi)$. Then $\ell \in Cl(\xi)$.
2) If $\xi = \heartsuit\chi$, then χ is tidy and $Cl(\xi) = \{\heartsuit\chi\} \cup Cl(\chi)$, where $\heartsuit \in \{\Diamond, \Box\}$.
3) If $\xi = \chi_0 \circledast \chi_1$ then both χ_i are tidy and $Cl(\xi) = \{\chi_0 \circledast \chi_1\} \cup Cl(\chi_0) \cup Cl(\chi_1)$, where $\circledast \in \{\wedge, \vee\}$.
4) If $\xi = \chi[\psi/x]$, χ is tidy, $x \in FV(\chi)$ and ψ is free for x in χ, then ψ is tidy and

$$Cl(\xi) = \{\varphi[\psi/x] \mid \varphi \in Cl(\chi)\} \cup Cl(\psi).$$

5) Let $\xi = \eta x.\chi$, where $\eta \in \{\mu, \nu\}$; assume that $x \in FV(\chi)$, and let x^* be some fresh variable. Then $\chi[x^*/x]$ is tidy and

$$Cl(\xi) = \{\varphi[\eta x.\chi/x^*] \mid \varphi \in Cl(\chi[x^*/x])\}. \tag{11}$$

Before we turn to the proof of Proposition 2.44, we briefly comment on the formulation of part 5). Note that if ξ is of the form $\xi = \eta x\, \chi$, then χ is not necessarily tidy, so that $Cl(\chi)$ may not be defined. For this reason we use a fresh propositional variable x^*. However, in case χ is tidy, (11) simplifies to

$$Cl(\xi) = \{\varphi[\eta x.\chi/x] \mid \varphi \in Cl(\chi)\}. \tag{12}$$

Proof. We prove the first and fourth claim of the proposition, leaving the other parts to the reader. The second and third claim are easy to prove, and part 5) is a fairly direct consequence of part 4).

For the first item, define the *height* of ℓ in ξ as the length of the shortest chain of the form $\varphi_0 \lhd_0 \varphi_1 \lhd_0 \cdots \lhd_0 \varphi_n$ such that $\varphi_0 = \ell$, $\varphi_n = \xi$, and, in case ℓ is a propositional variable p, no formula φ_i is of the form $\eta p\, \psi$. It is then straightforward to prove that $\ell \in Cl(\xi)$ by induction on the height of ℓ in ξ. We leave the details for the reader.

For the proof of 4), assume that $x \in FV(\chi)$ and that ψ is free for x in χ. By Proposition 2.42(3), the formula ψ is free for x in every $\varphi \in Cl(\chi)$. To prove the inclusion $\subseteq$ it suffices to show that the set $\{\varphi[\psi/x] \mid \varphi \in Cl(\chi)\} \cup Cl(\psi)\}$ has the required closure properties. This is easily verified, and so we omit the details.

For the opposite inclusion, we first show that

$$\varphi[\psi/x] \in Cl(\chi[\psi/x]), \text{ for all } \varphi \in Cl(\chi), \tag{13}$$

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

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

In the third case, we assume that $\varphi \in Cl(\chi)$ is of the form $\varphi = \rho[\lambda y.\rho/y]$, with $\lambda \in \{\mu, \nu\}$ and $\lambda y.\rho \in Cl(\chi)$. Then inductively we may assume that $(\lambda y.\rho)[\psi/x] \in Cl(\chi[\psi/x])$. Now we make a case distinction: if $x = y$ we find that $(\lambda y.\rho)[\psi/x] = \lambda y.\rho$, while at the same time we have $\varphi[\psi/x] = \rho[\lambda y.\rho/y][\psi/x] = \rho[\lambda y.\rho/y]$, so that it follows by the closure properties that $\varphi[\psi/x] \in Cl(\chi[\psi/x])$ indeed. If, on the other hand, x and y are distinct variables, then we find $(\lambda y.\rho)[\psi/x] = \lambda y.\rho[\psi/x]$, and so it follows by the closure properties that the formula $(\rho[\psi/x])\big[\lambda y.\rho[\psi/x]/y\big]$ belongs to $Cl(\chi[\psi/x])$. But since ψ is free for x in χ, the variable y is not free in ψ, and so a straightforward calculation shows that $(\rho[\psi/x])\big[\lambda y.\rho[\psi/x]/y\big] = \rho[\lambda y.\rho/y][\psi/x] = \varphi[\psi/x]$, and so we find that $\varphi[\psi/x] \in Cl(\chi[\psi/x])$ in this case as well. This proves (13).

To see why this implies part 4) of the proposition, it remains to show that $Cl(\psi) \subseteq Cl(\xi)$. But from $x \in FV(\chi)$ we infer $x \in Cl(\chi)$ by part 1), and from this we obtain that $\psi = x[\psi/x] \in Cl(\xi)$. This suffices by Proposition 2.43. QED

As an almost immediate corollary of Proposition 2.44 we find that the closure set of a μ-calculus formula is always *finite*.

Proposition 2.45 *Let $\xi \in \mu\mathtt{ML}$ be some formula. Then the set $Cl(\xi)$ is finite.*

Proof. We prove the proposition by induction on the *length* of a formula, as defined in Definition 2.5. More precisely, we claim that

$$|Cl(\xi)| \leq |\xi|^{\ell} \tag{14}$$

for every tidy formula $\xi \in \mu\mathtt{ML}$.

In case ξ is a formula of length 1 it must be atomic, so (14) is obvious. For the inductive case we consider a formula ξ with $|\xi|^{\ell} > 1$; such a formula cannot be atomic, and so it must be a boolean, modal or fixpoint formula. We now make a case distinction, only considering the cases where ξ is a conjunction or a μ-formula.

First let ξ be of the form $\xi = \xi_0 \wedge \xi_1$. By Proposition 2.44(3) we obtain $|Cl(\xi)| \leq |Cl(\xi_0)| + |Cl(\xi_1)|$, and the induction hypothesis yields $|Cl(\xi_i)| \leq |\xi_i|^{\ell}$. Thus we find $|Cl(\xi)| \leq |\xi_0|^{\ell} + |\xi_1|^{\ell} < |\xi|^{\ell}$.

If ξ is of the form $\xi = \mu x\, \chi$ we further distinguish cases. If x is not free in χ we have $\chi[\xi/x] = \chi$ and so $Cl(\xi) = \{\xi\} \cup Cl(\chi)$. Thus, using the induction hypothesis on χ, we obtain $|Cl(\xi)| \leq 1 + |Cl(\chi)| \leq 1 + |\chi|^{\ell} = |\xi|^{\ell}$, as required. On the other hand, if x does occur freely in χ, by Proposition 2.44(5) we find $|Cl(\xi)| \leq |Cl(\chi[x^*/x])|$. But since $\chi[x^*/x]$ has the same length as χ we may use the induction hypothesis for it; this gives $|Cl(\chi[x^*/x])| \leq |\chi[x^*/x]|^{\ell} = |\chi|^{\ell}$. Combining these observations we find that $|Cl(\xi)| \leq |\chi|^{\ell} = |\xi|^{\ell} - 1$ which obviously suffices to prove (14). QED

2.6 Measuring formulas

If we are interested in the complexity of algorithms for, e.g., model checking of a formula on a model, or satisfiability checking of a formula, we will see that two measures of a formulas feature prominently: its size and its alternation depth. Both notions are in fact quite subtle in that they admit several non-equivalent definitions.

Size

When it comes to *size*, there are at least three definitions that look reasonable, at first sight: in principle one could define the size of a formula as its *length*, its *subformula-size*, or its *closure-size*. Each definition corresponds to a certain way of *representing* a formula as a graph-based structure: the length of a formula corresponds to the number of nodes in its syntax tree, its subformula-size to the number of nodes in its subformula graph, and its closure-size to the size of its closure graph. In later chapters we will see that the main complexity results on μ-calculus formulas are actually proved using algorithms that operate on such graph-based representations. As a key example, the algorithm for *model checking* a μ-calculus formula in a pointed model is based on computing the winner of the associated evaluation game, and the two versions of the evaluation game that we have seen in this chapter crucially involve, respectively, the subformula graph and the closure graph of a formula. In other words, much more than just picking a number and referring to it as 'the size' of a formula, the more fundamental discussion concerns the various graph-based representations of a formula.

Definition 2.46 The *closure-size* $|\xi|$ of a tidy formula ξ is given by

$$|\xi|^c := |Cl(\xi)|,$$

i.e., it is defined as the number of formulas that are derived from ξ. The *subformula-size* of a clean formula ξ is defined as follows:

$$|\xi|^s := |Sf(\xi)|,$$

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

Note that while the notion of *length* applies to all formulas, this is different for the other two measures. In Remark 2.28 we saw that it only makes sense to define the subformula-based evaluation game for clean formulas. For this reason we restrict the definition of subformula-size to clean formulas, and for similar reasons we only define the closure-size for tidy formulas. These definitions could be extended to arbitrary formulas by considering alphabetical variants, but we shall have no need to do so.

In the remark below we discuss the advantages and disadvantages of the three different ways to represent μ-calculus formulas.

Remark 2.47 The tree representation and the associated concept of length are very useful when giving inductive definitions and proofs but not so convenient for algorithmic purposes because of the unnecessarily unwieldy size of trees. For instance, it is well-known that the

subformula-size of a formula can be exponentially smaller than its length, as is witnessed by the sequence of formulas $(\xi_n)_{n\in\omega}$ given by $\xi_0 := p$, $\xi_{n+1} := (\xi_n \wedge \xi_n)$. It is easy to see that the length of these formulas grows exponentially, whereas their subformula size grows linearly. For this reason in the sequel we leave the tree representations out of the picture and focus on a comparison between the subformula graph and the closure graph.

At first sight it seems more intuitive to work with subformulas because they are part of the formula itself, therefor always shorter, and they seamlessly fit with the natural inductive definition of formulas. Contrary to this, when defining the closure set of a formula, one easily get entangled in the complexities of performing substitutions which produce unreadably long formulas. In addition, proofs using the closure set can be more involved because we cannot always use straightforward inductive arguments.

Despite of this we argue that often the more natural representation of a μ-calculus formula is its closure graph, for the following reasons.

To start with, we already saw that working with subformulas makes sense for *clean* formulas only, while the closure set is well defined for all tidy formulas, which form a substantially larger class of formulas. Related to this but more importantly, recall that the equivalence of a fixpoint formula to its *unfolding* lies at the core of the semantics of the modal μ-calculus. The fact that the closure set of a formula is closed under taking unfoldings by design makes derived formulas more natural to work with than subformulas. For instance, this feature allows for a very smooth development of tableaux and derivation systems for $\mu\mathtt{ML}$ in Chapter 7, where the natural rule for fixpoint formulas simply means to unfold them.

Second, in Chapter 6 we will see that, perhaps counterintuitively, the closure graph of a formula can be *exponentially smaller* than its subformula graph. More precisely, Proposition 6.55 provides a sequence of formulas $(\xi_n)_{n\in\omega}$ of which the subformula-size grows exponentially, while the closure-size only grows linearly. Combining this with the above discussion one could say that

$$Sf(\xi) \text{ is a large set of short expressions, } Cl(\xi) \text{ is a small set of long expressions.}$$

When it comes to complexity issues, the point is that key algorithms such as model checking work as fast on the subformula graph as on the closure graph. This means that we obtain *sharper* results if we measure the size of a formula as its closure-size. ◁

Definition 2.48 The *size* of a tidy formula ξ is given as its closure-size: $|\xi| := |\xi|^c$. ◁

- ▶ `define size for arbitrary formulas?`
- ▶ `discuss size wrt alphabetical variance?`

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

Example 2.49 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 features 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 $\mathsf{ad}(\xi_2) = 2$ but $\mathsf{ad}(\xi_1) = 1$. $\lhd$

The formal definition of alternation depth involves inductively defined formula collections Θ_n^η, where $\eta \in \{\mu, \nu\}$ and n is a natural number. Intuitively, the class Θ_n^η 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. Recall our notation $\overline{\mu} = \nu$, $\overline{\nu} = \mu$.

Definition 2.50 By natural induction we define classes $\Theta_n^\mu, \Theta_n^\nu$ of μ-calculus formulas. With $\eta, \lambda \in \{\mu, \nu\}$ arbitrary, we set:

1. all atomic formulas belong to Θ_0^η;
2. if $\varphi_0, \varphi_1 \in \Theta_n^\eta$, then $\varphi_0 \vee \varphi_1, \varphi_0 \wedge \varphi_1, \Diamond\varphi_0, \Box\varphi_0 \in \Theta_n^\eta$;
3. if $\varphi \in \Theta_n^\eta$ then $\overline{\eta}x.\varphi \in \Theta_n^\eta$;
4. if $\varphi(x), \psi \in \Theta_n^\eta$, then $\varphi[\psi/x] \in \Theta_n^\eta$, provided that ψ is free for x in φ;
5. all formulas in Θ_n^λ belong to Θ_{n+1}^η.

The *alternation depth* $\mathsf{ad}(\xi)$ of a formula ξ is defined as the least n such that $\xi \in \Theta_n^\mu \cap \Theta_n^\nu$. A formula is *alternation free* if it has alternation depth at most 1. $\lhd$

Roughly, we obtain Θ_0^μ by closing the set of basic modal formulas under the boolean and modal operators, and the *greatest* fixpoint operator; and similarly for Θ_0^ν. Inductively, we obtain Θ_{n+1}^η by closing the set $\Theta_n^{\overline{\eta}}$ under the boolean and modal operations, substitution, and the $\overline{\eta}$-operator.

Remark 2.51 In the literature one usually sees the alternation hierarchy defined in terms of classes Σ_n and Π_n, with the notation taken from the arithmetical hierarchy. The connection with our notation is as follows:

$$\begin{array}{lcllcl} \Sigma_0 & := & \Theta_0^\mu \cap \Theta_0^\nu & \Sigma_{n+1} & := & \Theta_n^\nu \\ \Pi_0 & := & \Theta_0^\mu \cap \Theta_0^\nu & \Pi_{n+1} & := & \Theta_n^\mu. \end{array}$$

Our notation, which uses μ and ν as superscripts, allows to exploit the symmetry between μ and ν more directly, and may thus make the definition of alternation depth a bit easier. $\lhd$

Example 2.52 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_0^ν but not to Θ_0^μ: none of the clauses in Definition 2.50 is applicable. On the other hand, using clause (5) it is easy to see that $\mu x.p \vee x \in \Theta_1^\nu \cap \Theta_1^\mu$, from which it is immediate that $\mathsf{ad}(\mu x.p \vee x) = 1$.

Returning to Example 2.49, consider the formula $\xi_1 = \mu x.(\nu y.p \wedge \Box y) \vee \Diamond x$. Taking a fresh variable q, we find $\mu x.q \vee \Diamond x \in \Theta_0^\nu \subseteq \Theta_1^\nu$ and $\nu y.p \wedge \Box y \in \Theta_0^\mu \subseteq \Theta_1^\nu$, so that by the substitution rule we have $\xi_1 = (\mu x.q \vee \Diamond x)[\nu y.p \wedge \Box y/q] \in \Theta_1^\nu$. Similarly we may show that $\xi_1 \in \Theta_1^\mu$, 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_0^ν but not to Θ_0^μ. From this it follows that ξ_2 belongs to Θ_1^μ but there is no way to place it in Θ_1^ν. Hence we find that $\mathsf{ad}(\xi_2) = 2$.

As a final 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 the variable z is not dependent on y. Consequently 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 $\mathsf{ad}(\xi_3) = 2$. $\lhd$

In Chapter 6 we will make precise the link between the formally defined alternation depth of a formula and the more intuitive notion of alternating chains of bound variables (in the case of clean formulas) and of fixpoint formulas (in the case of tidy formulas).

Remark 2.53 A natural question is whether the alternation *hierarchy* is *strict*; that is, whether there are formulas of arbitrary alternation depth. The (affirmative) answer to this question will be given in Chapter 11. $\lhd$

We finish our discussion of alternation depth here by mentioning that each class Θ_n^η is closed under taking subformulas and derived formulas.

Proposition 2.54 *Let ξ and χ be μ-calculus formulas. Then the following hold:*

$$\text{if } \xi \in \Theta_n^\eta \text{ and } \varphi \trianglelefteq \xi \quad \text{then } \varphi \in \Theta_n^\eta \tag{15}$$

$$\text{if } \xi \in \Theta_n^\eta \text{ and } \xi \twoheadrightarrow_C \varphi \quad \text{then } \varphi \in \Theta_n^\eta \tag{16}$$

As a corollary, we have $\mathsf{ad}(\chi) \leq \mathsf{ad}(\xi)$ if ξ is clean and $\chi \in Sf(\xi)$, or ξ is tidy and $\chi \in Cl(\xi)$.

Proof. The statement in (15) can be proved by a straightforward induction on the derivation of $\xi \in \Theta_n^\eta$ — we leave the details for the reader. •

To prove (16), it suffices to show that the class Θ_n^η is closed under unfoldings, since by (15) 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 (15) that $\chi \in \Theta_n^\lambda$. But then we may apply clause (4) from Definition 2.50 and conclude that $\chi[\eta.\chi/x] \in \Theta_n^\lambda$. QED

2.7 Substitutions and free subformulas

In this section we discuss some further syntactic issues related to the modal μ-calculus. None of these is of high intrinsic importance, but all concepts and results that we introduce or prove here will play a role in the development of the general theory.

Substitution lemmas

We start with proving some lemmas on substitution. The following proposition is a well known observation in areas where syntax is used that features variable binding. Note however that our version below is a bit subtler than usual since we do not allow the renaming of bound variables.

Proposition 2.55 *Let φ, χ and ρ be μ-calculus formulas, and let x and y be distinct variables such that x is free in φ but not in ρ. Furthermore, assume that χ is free for x in φ and that ρ is free for y in $\varphi[\chi/x]$. Then ρ is free for y in both φ and χ, $\chi[\rho/y]$ is free for x in $\varphi[\rho/y]$, and we have*

$$\varphi[\chi/x][\rho/y] = \varphi[\rho/y][(\chi[\rho/y])/x]. \tag{17}$$

Proof. The proposition can be proved by a straightforward but rather tedious induction on the complexity of φ. We omit details. QED

A major theme in this book is to study representations of formulas as *structures*, that is, sets with additional relations and operations. It will be of interest to study the properties of substitutions seen as maps between such structures. The next proposition states that substitutions have the characteristic property of bounded morphisms, with respect to the trace relation.

Proposition 2.56 (Back- and forth property of substitution) *Let ξ and χ be tidy μ-calculus formulas such that $\chi[\xi/x]$ is tidy. Then the substitution operation $\xi/x : Cl(\chi) \to \mu\mathtt{ML}$ satisfies the following back- and forth condition, for every $\varphi \in Cl(\chi) \setminus \{x\}$:*

$$\{\chi \mid \varphi[\xi/x] \to_C \chi\} = \{\psi[\xi/x] \mid \varphi \to_C \psi\}. \tag{18}$$

Proof. We distinguish cases depending on the shape of φ.

If $\varphi \neq x$ is atomic then there are no ψ with $\varphi \to_C \psi$. Because there is also no $\to_C$-successor of $\varphi[\xi/x] = y[\xi/x] = \varphi$ the claim holds trivially.

The cases where $\varphi = \varphi_0 \odot \varphi_1$ with $\odot \in \{\wedge, \vee\}$, or $\varphi = \heartsuit\psi$ with $\heartsuit \in \{\Diamond, \Box\}$ are straightforward.

If $\varphi = \eta y.\rho$, then the unique $\to_C$-successor of φ is the formula $\rho[\varphi/y]$. We distinguish further cases depending on whether $y = x$. If $y = x$ then we have $\varphi[\xi/x] = \varphi$, and this formula has again $\rho[\varphi/y]$ as its only $\to_C$-successor. Because $x \notin FV(\varphi)$ we also have $\rho[\varphi/x][\xi/x] = \rho[\varphi/x]$ and thus the claim holds trivially.

If $y \neq x$ then we find $\varphi[\xi/x] = \eta y.\rho[\xi/x]$, and the unique $\to_C$-successor of this formula is its unfolding $\big(\rho[\xi/x]\big)[\varphi[\xi/x]/y]$. For the right hand side of (18), obviously the unique $\to_C$-successor of φ is its unfolding $\rho[\varphi/y]$. It is thus left to show that

$$\big(\rho[\xi/x]\big)[\varphi[\xi/x]/y] = \big(\rho[\varphi/y]\big)[\xi/x]. \tag{19}$$

But we have $BV(\varphi) \subseteq BV(\chi)$ by Proposition 2.42 and $BV(\chi) \cap FV(\xi) \subseteq \{x\}$ by the tidyness of $\chi[\xi/x]$. Thus it follows from $y \in BV(\varphi)$ and $y \neq x$ that $y \notin FV(\xi)$. Hence (19) follows by Proposition 2.55 — the condition that ξ is free for x in $\rho[\varphi/y]$ follows by Proposition 2.42 as well. QED

Free subformulas

We now have a closer look at the relation between the sets $Sf(\xi)$ and $Cl(\xi)$. 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 2.57 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'. $\lhd$

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 $\psi \in Cl(\varphi)$, or equivalently, if there is a trace (possibly of length zero) from φ to ψ.

Proposition 2.58 *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 $\lhd_0$-chain $\varphi = \chi_0 \lhd_0 \chi_1 \lhd_0 \cdots \lhd_0 \chi_n = \psi$, 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 $\lhd_0$-chain $x = \chi'_0 \lhd_0 \chi'_1 \lhd_0 \cdots \lhd_0 \chi'_n = \psi'$ 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 $\lhd_0$-chain $\varphi = \chi_0 \lhd_0 \chi_1 \lhd_0 \cdots \lhd_0 \chi_n = \psi$ 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 $\lhd_0$-chain $\varphi = \chi_0 \lhd_0 \chi_1 \lhd_0 \cdots \lhd_0 \chi_n = \psi$. 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) ⇒ 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 $\rho_0 \circledast \rho_1$ with $\circledast \in \{\wedge, \vee\}$, 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 $\chi_{n+1} \to_C \chi_n$ 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 $\chi_{n+1} \to_C \chi_n[\chi_{n+1}/y]$ we find that $\chi_{n+1} \twoheadrightarrow_C \chi_0$ indeed. This finishes the proof of the implication 4) ⇒ 3).

Finally, it follows from Proposition 2.42(1) that $\psi \twoheadrightarrow_C \varphi$ implies $FV(\varphi) \cap BV(\psi) \subseteq FV(\psi) \cap FV(\psi) = \varnothing$. From this the implication 3) ⇒ 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.

Proposition 2.59 *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*

$$\varphi_0[\xi/x] = \varphi_1[\xi/x] \text{ implies } \varphi_0 = \varphi_1. \tag{20}$$

Proof. We first observe that, with $\varphi_0, \varphi_1, \xi$ and x as in the statement of the proposition, it holds that

$$\varphi_0[\xi/x] = \varphi_1[\xi/x] \text{ implies that } x \in FV(\varphi_0) \text{ iff } x \in FV(\varphi_1). \tag{21}$$

This is in fact easy to see: if $x \in FV(\varphi_0) \setminus FV(\varphi_1)$, then we would obtain a contradiction from $\xi \trianglelefteq_f \varphi_0[\xi/x] = \varphi_1[\xi/x] = \varphi_1$.

We now turn to (20), which we will prove by a straightforward induction on the complexity of φ_0. Note that by (21) we only need to worry if $x \in FV(\varphi_0) \cap FV(\varphi_1)$; if this is not the case then (20) holds trivially.

In particular, this means that the base step of the inductive proof of (20) is reduced to the case where $\varphi_0 = \varphi_1 = x$, so that (20) holds trivially.

For the inductive step we first consider the case where φ_0 is of the form $\psi_0 \wedge \chi_0$. Then we obtain $\varphi_0[\xi/x] = \psi_0[\xi/x] \wedge \chi_0[\xi/x]$. But if $\varphi_1[\xi/x] = \psi_0[\xi/x] \wedge \chi_0[\xi/x]$ and $\xi \not\trianglelefteq_f \varphi_1$, it must be the case that φ_1 is of the form $\varphi_1 = \psi_1 \wedge \chi_1$, with $\psi_1[\xi/x] = \psi_0[\xi/x]$ and $\chi_1[\xi/x] = \chi_0[\xi/x]$. By the induction hypothesis we obtain $\psi_0 = \psi_1$ and $\chi_0 = \chi_1$, so that $\varphi_0 = \varphi_1$ indeed.

The cases for disjunction and the modal operators are very similar to this, so we omit the details.

This leaves the case where φ_0 is of the form $\varphi_0 = \eta y.\psi_0$. Restricting to the case where $x \in FV(\varphi_0) \cap FV(\varphi_1)$ we may assume that x and y are distinct variables, so we have $\varphi_0[\xi/x] = \eta y.\psi_0[\xi/x]$. But it follows from $\eta y.\psi_0[\xi/x] = \varphi_1[\xi/x]$ and $\xi \not\trianglelefteq_f \varphi_1$, that φ_1 must be of the form $\varphi_1 = \eta y.\psi_1$ for some formula ψ_1 with $\psi_0[\xi/x] = \psi_1[\xi/x]$. Thus the inductive hypothesis yields that $\psi_0 = \psi_1$, which immediately implies that $\varphi_0 = \varphi_1$ as required. QED

The expansion map

The most important observation here concerns the existence of a surjective map from $Sf(\xi)$ to $Cl(\xi)$, 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 $y \trianglelefteq \delta_x$.

Definition 2.60 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 2.61 *Let $\xi \in \mu\mathtt{ML}$ be a clean formula and $\mathbb{S}$ a pointed Kripke structure. Then for all subformulas $\varphi \trianglelefteq \xi$ and all states s in $\mathbb{S}$ we have*

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

The proposition below states that, for a clean formula ξ, the expansiom map is a surjection from its set of subformulas of ξ to its closure. As an immediate corollary we obtain that the size of $Cl(\xi)$ is *bounded* by that of $Sf(\xi)$.

Proposition 2.62 *Let ξ be a clean $\mu\mathtt{ML}$-formula. Then*

$$Cl(\xi) = \{\exp_\xi(\varphi) \mid \varphi \trianglelefteq \xi\}. \tag{22}$$

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 $\exp_\xi(\varphi) \in Cl(\xi)$, for every $\varphi \in Sf(\xi)$, which can be done by a straightforward induction. QED •

Notes

The modal μ-calculus was introduced by D. Kozen [10]. Its game-theoretical semantics goes back to at least Emerson & Jutla [6] (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 [11].

There are various ways to make the notion of alternation depth precise; we work with the most widely used definition, which originates with Niwiński [16].

▶ `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 $s_0 \stackrel{d_0}{\to} s_1 \stackrel{d_1}{\to} s_2 \cdots$ 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, 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 (traces)

(a) Let $\xi = \eta x\, \chi$ be a tidy fixpoint formula, and let φ be a subformula of its unfolding. Show that either $\varphi \trianglelefteq \xi$ or $\xi \trianglelefteq \varphi$.

Hint: prove that every subformula $\alpha \trianglelefteq \chi$ satisfies the following:

$$\text{if } \varphi \trianglelefteq \alpha[\xi/x] \text{ then } \varphi \trianglelefteq \xi \text{ or } \xi \trianglelefteq \varphi.$$

(b) Let $\tau : \xi_n \to_C \ldots \to_C \xi_1$ be a finite trace of tidy formulas. Show that there is a unique $\xi \in \{\xi_1, \ldots, \xi_n\}$ such that $\xi \trianglelefteq \xi_i$, for all i. Moreover, if ξ_1 is a fixpoint formula then so is ξ.

(c) Let τ be an infinite trace of tidy formulas, and let $\mathit{Inf}(\tau)$ be the set of formulas that occur infinitely often on τ. Show that there is a unique formula $\xi \in \mathit{Inf}(\tau)$ which is a subformula of every formula in $\mathit{Inf}(\tau)$. Furthermore, prove that ξ is a fixpoint formula.

Exercise 2.10 (injectivity of substitution) Prove Proposition 2.59.

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] J. van Benthem. *Modal Correspondence Theory*. PhD thesis, Mathematisch Instituut & Instituut voor Grondslagenonderzoek, University of Amsterdam, 1976.

[3] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Number 53 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.

[4] 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.

[5] A. Chagrov and M. Zakharyaschev. *Modal Logic*, volume 35 of *Oxford Logic Guides*. Oxford University Press, 1997.

[6] 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.

[7] 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.

[8] 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.

[9] B. Knaster. Un théorème sur les fonctions des ensembles. *Annales de la Societé Polonaise de Mathematique*, 6:133–134, 1928.

[10] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[11] 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.

[12] R. McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and Control*, 9:521–530, 1966.

[13] L. Moss. Coalgebraic logic. *Annals of Pure and Applied Logic*, 96:277–317, 1999. (Erratum published *Ann.P.Appl.Log.* 99:241–259, 1999).

[14] 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.

[15] 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.

[16] 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.

[17] D. Park. Concurrency and automata on infinite sequences. In *Proceedings 5th GI Conference*, pages 167–183. Springer, 1981.

[18] A. Pnueli. The temporal logic of programs. In *Proc. 18th Symp. Foundations of Computer Science*, pages 46–57, 1977.

[19] V.R. Pratt. Semantical considerations on Floyd-Hoare logic. In *Proc. 17th IEEE Symposium on Computer Science*, pages 109–121, 1976.

[20] 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.

[21] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[22] I. Walukiewicz. Completeness of Kozen's axiomatisation of the propositional μ-calculus. In *Proceedings of the 10th Annual IEEE Symposium on Logic in Computer Science (LICS'95)*, pages 14–24. IEEE Computer Society Press, 1995.

[23] I. Walukiewicz. Completeness of Kozen's axiomatisation of the propositional μ-calculus. *Information and Computation*, 157:142–182, 2000.

[24] W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200:135–183, 1998.