Formal Systems 2

P. H. Schmitt

May 2016

Contents

List of Figures

Chapter 1

Introduction

1.1 Prerequisites

We assume that the reader has had some previous exposure to propositional and first-order logic. The lecture notes [Schmitt, 2008] will certainly be good enough. But, also any good textbook on the subject [Fitting, 1990, Monk, 1976, Huth & Ryan, 2000] will do.

Here we will occupy ourselves with some interesting yet not mainstream results to get back into the mood. The emphasis is on providing exercises to work on.

Meagre Vocabularies

To get a better insight into first-order logic logicians have taken a great interest in studying the expressive power of various fragments of first-order formulas. Well know examples are the fragments of Horn formulas, existential or universal formulas or quantifier-free formulas. Here will will not restrict the use of the logical operations, but put limitations on the vocabularies. It is for example well know that it is no restricition to use only relation symbols and forget function symbols of arity strictly greater than 0.

Definition 1 (Vocabularies)
A vocabulary Σ *is called*

1. *a* relational vocabulary *if it possibly contains constant symbols, but no function symbols of arity greater than* 0.
2. *a* binary vocabulary *if it is relational and all relations symbols in* Σ *are binary.*
3. *a* triple vocabulary *if it is relational and contains exactly on relation symbol rel and this is ternary.*

We also need a way to compare different types of vocabularies with regard to their expressiveness.

Definition 2 (Comparing Vocabularies)
Let $\mathcal{V}_1$, $\mathcal{V}_2$ *be two classes of vocabularies.* $\mathcal{V}_2$ *is called as expressive as* $\mathcal{V}_1$ *if for every signature* Σ_1 *in class* $\mathcal{V}_1$ *there is a signature* Σ_2 *in class* $\mathcal{V}_2$ *such that*

1. *for every* Σ_1*-formula* F_1 *there is a* Σ_2*-formula* F_2 *and for every* Σ_2*-structure* $\mathcal{M}$ *there is a* Σ_1*-structure* $\mathcal{M}_1$ *such that*

$$\mathcal{M}_1 \models F_1 \Leftrightarrow \mathcal{M} \models F_2$$

2. *and for every* Σ_1*-structure* $\mathcal{N}$ *there is a* Σ_2*-structure* $\mathcal{M}$ *such that*

$$\mathcal{M}_1 \simeq \mathcal{N}$$

This definition may at first sound complicated. But, in the end it is just a formal rendering of what you would intuitivly assume. To get some understanding for this definition try to do excercise 1.2.2.

Definition 2 describes the ideal situation. In most cases, in particular also for Exercises 1.2.4 and 1.2.5, the transition from $\mathcal{M}$ to $\mathcal{M}_1$ is only possible for structures $\mathcal{M}$ that are big enough.

Order Relations

Definition 3 (Order) *A structure* (M, R) *where* R *is a binary relation on* M *is called an* order *if it satisfies the following axioms*

1. $\forall x \forall y \forall z (R(x, y) \wedge R(y, z) \rightarrow R(x, z))$ *(transitivity)*
2. $\forall x (\neg R(x, x))$ *(anti-reflexivity)*

Definition 4 (Linear Order) *A order relation* (M, R) *is called a* linear order *if it satisfies the following axiom*

1. $\forall x \forall y \forall z (R(x, y) \vee R(y, x) \vee x = y)$ *(linearity)*

1.2 Exercises

Exercise 1.2.1
Let F be a formula in first-order logic, that is true in all infinite structures. Then there is a natural number k such that F is even true in all structures with k or more elements.

Exercise 1.2.2
Assume that Σ_1, Σ_2, F_1, F_2 satisfy the two statements from Definition 2 on page 7.
Then F_1 is a Σ_1-tautology iff F_2 is a Σ_2-tautology.

Exercise 1.2.3
Assume that Σ_1, Σ_2, F_1, F_2 satisfy the two statements from Definition 2 on page 7.
If validity of Σ_1-formulas is undecidable then also validity of Σ_2-formulas is undecidable.

Exercise 1.2.4
Show that the class of triple vocabularies is as expressive as the class of binary vocabularies.

Exercise 1.2.5
Show that the class of binary vocabularies is as expressive as the class of unrestricted relational vocabularies.

See also Exercise 3.9.23.

Chapter 2

Axiomatic Set Theory

Many formal specification languages, among them as prime examples **Z** and **B**, use set theoretical concepts and notations. This is an appealing choice, because these concepts are easy to understand and accessible without mathematical training. Another advantage is the fact, that there is a well developed mathematical theory of sets. In fact, before set theory was perceived as a foundation for specification languages it was considered as a foundation for all of mathematics. A very intriguing idea: once you accept a few axioms of set theory all mathematical results can be derived from them. In this chapter we will convey a first idea of how this works.

2.1 Basics

We will use the Zermelo-Fraenkel (ZF for short) axiom system for set theory. In our presentation we follow the textbooks [Takeuti & Zaring, 1971], [Rubin, 1967], and [R.Drake, 1974].

The full set of ZF axioms is given in Figure 2.1 on page 36.

The language of ZF set theory is the language for first-order predicate logic with the binary relation symbol $\in$ as its only non-logical symbol. In the formulation of the axioms the equality symbol $=$ is also used. But note, that using axiom **A1** and formula $t_1 = t_2$ may be equivalently replaced by a formula containing only $\in$. More precisely, axiom **A1** states only one implication. The reverse implication, i.e.

$$x = y \rightarrow \forall z(z \in x \leftrightarrow z \in y)$$

has nothing to do with set theory, it is a simple consequence of the congruence axioms

$$x = y \rightarrow (p(z, x) \leftrightarrow p(z, y))$$

for any binary relation symbol p.

Any free variables in the axioms are implicitely universally quantified.

Before we go on, we need some notational conventions, otherwise our formulas would soon be unintelligible.

We will use for any formula $\phi(x)$ the syntactical construct $\{x \mid \phi(x)\}$, called a class term. We intuitively think of $\{x \mid \phi(x)\}$ as the collection of all sets

a satisfying the formula $\phi(a)$. This is only for notational convenience. The new terms can be eliminated as follows:

$y \in \{x \mid \phi(x)\}$	is replaced by	$\phi(y)$
$\{x \mid \phi(x)\} \in y$	is replaced by	$\exists u(u \in y \wedge$ $\forall z(z \in u \leftrightarrow \phi(z)))$
$\{x \mid \phi(x)\} \in \{y \mid \psi(y)\}$	is replaced by	$\exists u(\psi(u) \wedge$ $\forall z(z \in u \leftrightarrow \phi(z)))$

Note, that using a class term $\{x \mid \phi(x)\}$, does by far not imply that $\{x \mid \phi(x)\}$ is a set. For $\phi(x) := x \notin x$ this would immediately result in a contradiction. Only after we can prove that $\exists y(y = \{x \mid \phi(x)\})$ can be derived from the axioms, can we use $\{x \mid \phi(x)\}$ as a set.

Having class terms is already very handy, but still further abbreviations are necessary. Here is the first bunch:

Definition 5 (Abbreviations for sets)

$\emptyset$	$=$	$\{x \mid x \neq x\}$	
$\{a, b\}$	$=$	$\{x \mid x = a \vee x = b\}$	
$\{a\}$	$=$	$\{x \mid x = a\}$	
$\langle a, b\rangle$	$=$	$\{\{a\}, \{a, b\}\}$	*This is called the ordered pair of a and b*

Note, that some of these abbreviations have already been used in the axioms in Figure 2.1.

Let us look at some easy logical derivations from the ZF axioms.

Lemma 1 *The following formulas follow from the ZF axioms*

1. $\exists x(x = \emptyset)$
2. $\forall x, y \exists z(z = \{x, y\})$
3. $\forall x \exists z(z = \{x\})$
4. $\forall x, y \exists z(z = \langle x, y\rangle)$

Proof: The first step in all four proofs will be to unfold the abbreviating notation of class terms.

1. In a first step we eliminate the $=$ symbol in $\exists x(x = \emptyset)$ using the extensionality axiom, which yields: $\exists x \forall u(u \in x \leftrightarrow u \in \emptyset)$. Now the class term $\emptyset$ is replaced as explained above: $\exists x \forall u(u \in x \leftrightarrow u \neq u)$. Since $u \neq u$ is contradictory, this is equivalent to $\exists x \forall u(u \in x \rightarrow u \neq u)$. Which is logically equivalent to $\exists x \forall u(u \notin x)$, and this is Axiom **A4**.

2. Eliminating $=$ and the class term in $\forall x, y \exists z(z = \{x, y\})$ yields $\forall x, y \exists z \forall u(u \in z \leftrightarrow u = x \vee u = y)$. This is, after renaming of variables, axiom **A5**.

3. Special case of 2.

4. Unfolding the definition of an ordered pair, we get $\forall x, y \exists z(z = \{\{x\}, \{x, y\}\})$.
   In first-order logic
   $$\frac{\forall w \psi}{\forall \vec{w} \psi[t/w]}$$
   is a valid rule, where $\psi[t/w]$ results from ψ by replacing all free occurences of the variable w by the term t and $\vec{w}$ are all variables in t. Can this also be done with class terms? In general the answer is, no. But, if we can prove for a class term ct $\forall \vec{w} \exists u(u = ct)$, then the same replacement principle is true in ZF. Now, claim 4 follows from 2 and 3.

∎

Lemma 2

1. *If a and b are sets, then there is a set c satisfying*
   $$\forall z(z \in c \leftrightarrow z \in a \wedge z \in b)$$
   c is called the intersection of a and b, in symbols $c = a \cap b$.

2. *If a and b are sets, then there is a set c satisfying*
   $$\forall z(z \in c \leftrightarrow z \in a \vee z \in b)$$
   c is called the union of a and b, in symbols $c = a \cup b$.

3. *If A is a non-empty class term, then there is a set c satisfying*

$$\forall z(z \in c \leftrightarrow \forall u(u \in A \rightarrow z \in u))$$

c is called the intersection of A, in symbols $c = \bigcap A$.

4. *If a is a set, then there is a set c satisfying*

$$\forall z(z \in c \leftrightarrow \exists u(u \in a \wedge z \in u))$$

c is called the union of a, in symbols $c = \bigcup a$.

Proof: Let us for the moment be pedantic.

1. This requires the subset axiom A3

$$\exists y \forall z(z \in y \leftrightarrow z \in x \wedge \phi(z)).$$

We replace the free variable x by a, the formula $\phi(z)$ by $z \in b$ and name the element whose existence is guaranteed by the axiom c. This leads to

$$\forall z(z \in c \leftrightarrow z \in a \wedge z \in b)$$

as required.

2. Despite the fact that set theoretical union is such a simple concept, it does need two axioms to guarantee its existence. From the pair axioms, A5, we get the existence of a the set $d = \{a, b\}$ and the sum axiom, A7 yields the existence of a set c satisfying

$$\forall z(z \in c \leftrightarrow \exists u(u \in d \wedge z \in u))$$

Substituting $d = \{a, b\}$ yields the claim.

3. Let $A = \{u \mid \psi(u)\}$. Since A is assumed to be non-empty, we may pick an arbitrary element $b \in A$, i.e. an arbitrary b such that $\psi(b)$ is true. Let $\phi(z)$ be the formula $\forall u(\psi(u) \rightarrow z \in u)$. We will, again, use Axiom A3

$$\exists y \forall z(z \in y \leftrightarrow z \in b \wedge \phi(z)).$$

The element, whose existence is guaranteed is named c. This yields the claim, when we observe the trivial equivalence

$$\forall z \forall u(\psi(u) \rightarrow z \in u) \leftrightarrow z \in b \wedge \forall u(\psi(u) \rightarrow z \in u)$$

4. Use axiom A7.

∎

Likewise it is easy to prove

Lemma 3
$\forall x_1, x_2, y_1, y_2(\quad \langle x_1, x_2 \rangle = \langle y_1, y_2 \rangle \leftrightarrow x_1 = y_1 \wedge x_2 = y_2 \quad)$

Definition 6

1. *A relation r is a set of ordered pairs, i.e.*
   $rel(r) \equiv \forall x(x \in r \rightarrow \exists x_1, x_2(x = \langle x_1, x_2 \rangle))$
2. *The relation r is said to be a relation on the set s if*
   $rel(r, s) \equiv rel(r) \wedge \forall x_1, x_2(\langle x_1, x_2 \rangle \in r \rightarrow x_1 \in s \wedge x_2 \in s)$
3. *A function is a* one-valued *relation, i.e.*
   $func(r) \equiv rel(r) \wedge \forall x, y_1, y_2(\langle x, y_1 \rangle \in r \wedge \langle x, y_2 \rangle \in r \rightarrow y_1 = y_2)$
4. *A function f is said to be a function from a set a to a set b if*
   $func(f, a, b) \equiv func(f) \wedge \forall x_1, x_2(\langle x_1, x_2 \rangle \in f \rightarrow x_1 \in a \wedge x_2 \in b)$

Lemma 4
From the ZF axioms we can prove for any sets a, b the existence of the set of all relations on a and of all functions from a to b, i.e.

1. $\forall x \exists y \forall z(z \in y \leftrightarrow rel(z, x))$
2. $\forall u, w \exists y \forall z(z \in y \leftrightarrow func(z, u, w))$

Proof: For this proof we need (for the first time in this text) the power set axiom $\exists y \forall z(z \in y \leftrightarrow \forall u(u \in z \rightarrow u \in x))$. We denote the set whose existence is stipulated by this axiom by $\mathcal{P}(x)$.

1. For $x, y \in a$ the ordered pair $\langle x, y \rangle$ is an element of $\mathcal{P}(\mathcal{P}(a))$. The class term $Rel(a)$ of all relations on a is thus a subset of $\mathcal{P}(\mathcal{P}(\mathcal{P}(a))) = \mathcal{P}^3(a)$. The power set axiom tells us that $\mathcal{P}^3(a)$ exists. Since we can describe by a first-order formula exactly which elements of $\mathcal{P}^3(a)$ belong to $Rel(a)$ we get its existence by the subset axiom.

2. Similar.

∎

We will have occasion to use the following, well known, definitions:

Definition 7 *Let r be a relation.*

1. *the set* $\mathrm{dom}(r) = \{x \mid \exists y(\langle x, y\rangle \in r)\}$ *is called the domain of* r
2. *the set* $\mathrm{ran}(r) = \{x \mid \exists y(\langle y, x\rangle \in r)\}$ *is called the range of* r

If r is furthermore a function and $a \in \mathrm{dom}(r)$

3. *$r(a)$ is the unique element b satisfying $\langle a, b\rangle \in r$.*
   Note that we can write $r(a)$ as the class term $r(a) = \bigcup\{y \mid \langle a, y\rangle \in r\}$.

2.2 The Natural Numbers

Definition 8 (Successor) *For any set a the set*

$$a^+ = a \cup \{a\}$$

is called the successor set of a.

From our previous results it is obvious that a^+ is a set, when a is. In the following we will no longer mention facts of this simple kind explicitely.

We will use the empty set $\emptyset$ to represent the natural number 0, $\emptyset^+ = \{\emptyset\} = \{0\}$ to represent 1, $1^+ = \emptyset^{++} = \{\emptyset, \{\emptyset\}\} = \{0, 1\}$ to represent 2. In general, for any natural number n we let n^+ represent its successor. We want the set of natural numbers to be $\mathbb{N} = \{0, 0^+, 0^{++}, 0^{+++}, \ldots\}$. It remains to be explain how this can be turned into a legal definition and prove the existence of $\mathbb{N}$ from the ZF axioms.

Definition 9 *A set a is called a Dedekind set if $0 \in a$ and for all $b \in a$ also $b^+ \in a$. In symbols $Ded(a) \equiv 0 \in a \wedge \forall x(x \in a \to x^+ \in a)$.*

Lemma 5

$$\exists y(y = \bigcap\{a \mid Ded(a)\}$$

can be derived from the ZF axioms.

$\bigcap\{a \mid Ded(a)\}$ *will be called the set of natural numbers and denoted by* $\mathbb{N}$. *In set theory it is also customary to use the symbol* ω *instead of* $\mathbb{N}$.

Proof: The claim follows from Lemma 2(3) if we can show that there is at least one set a with $Ded(a)$. But this is guaranteed by Axiom A8, the infinity axiom.

∎

The Peano axiom system is usually taken as an axiomatic characterisation of the natural numbers. In the context of set theory we should be able to derive them from the set theoretical axioms.

Lemma 6 (Peano Arithmetic) *The following theorems can be derived from the ZF axioms*

1. $0 \in \mathbb{N}$.
2. *If* $n \in \mathbb{N}$ *then* $n^+ \in \mathbb{N}$.
3. $\forall n(n \in \mathbb{N} \to n^+ \neq 0)$.
4. $\forall n, m(n \in \mathbb{N} \wedge m \in \mathbb{N} \wedge n^+ = m^+ \to n = m)$.
5. $\forall x(0 \in x \wedge \forall y(y \in x \to y^+ \in x) \to \mathbb{N} \subseteq x)$.

Proof: 1 and 2 are obvious by definition of $\mathbb{N}$.

To prove 3 we note that $n \in n^+$ is true for any n, thus n^+ cannot be the empty set.

Assume for a proof of 4 that $n^+ = m^+$, i.e. $n \cup \{n\} = m \cup \{m\}$. Thus we must have

1. $m \in n \cup \{n\}$, i.e. $n = m$ or $m \in n$.

2. $n \in m \cup \{m\}$, i.e. $n = m$ or $n \in m$.

The foundation axiom, A2,

$$\exists y(y \in x) \to \exists y(y \in x \wedge \forall z \neg(z \in x \wedge z \in y)),$$

instantiated for $x = \{n, m\}$ yields after some simplifications $n \notin m$ or $m \notin n$. Thus the above case distinction forces $n = m$.

Part 5 is again simple. Any x satisfying the premise of the implication is a Dedekind set. Since $\mathbb{N}$ is by definition the intersection of all Dedekind sets, we obviously get $\mathbb{N} \subseteq x$.

∎

Before we move on to other topics let us pause to look at set theoretic properties of $\mathbb{N}$, i.e. properties that are not part of our intuition of natural numbers but arise from the particular set theoretic construction we have used to arrive at $\mathbb{N}$. We begin with a simple exercise on induction.

Lemma 7
For all $n \in \mathbb{N}$ with $n \neq 0$

$$0 \in n$$

Proof Set

$$x = \{n \in \mathbb{N} \mid 0 \in n\} \cup \{0\}$$

We need to show $x = \mathbb{N}$. This is best done using the induction axiom. To apply it we need to establish $0 \in x$, which is simply true by definition of x and $n \in x \to n^+ \in x$. There are two cases to be distinguished

Case $n \neq 0$ in this case $n \in x$ implies $0 \in n$. Since $n \subseteq n^+ = n \cup \{n\}$ we also get $0 \in n^+$ and thus $n^+ \in x$.

Case $n = 0$ Here $n^+ = \{0\}$ by definition. Thus obviously $0 \in n^+$ and also $n^+ \in x$.

∎

The element relation $\in$ is in general far from transitive, i.e. $a \in b$ and $b \in c$ do not entail $a \in c$. There are however interesting special situations where this is nevertheless true. This leads to the following definition

Definition 10 (Transitive Set)
A set a is called transitive if the $\in$-relation restricted to a is a transitive relation. In symbols

$$trans(a) \leftrightarrow \forall x(x \in a \rightarrow x \subseteq a)$$

Do not confuse transitive sets with transitive relations.

Lemma 8

1. *n is transitive for all $n \in \mathbb{N}$.*
2. *$\mathbb{N}$ is transitive.*

Proofs

Ad 1 We use induction. The empty set 0 is transitive by definition. Assume n is transitive and consider $x \in n^+ = n \cup \{n\}$ with the aim to show $x \subseteq n^+$. If $x \in n$ then by hypothesis $x \subseteq n \subseteq n^+$. If $x = n$, then by definition $x \subseteq n^+$.

Ad 2 We prove $\forall n(n \in \mathbb{N} \rightarrow n \subseteq \mathbb{N})$ again by induction. For $n = 0$ this is clear since the empty set is a subset of every set. If $n \in \mathbb{N}$ and by induction hypothesis $n \subseteq \mathbb{N}$ then also $n^+ = n \cup \{n\} \subseteq \mathbb{N}$.

∎

The order relation $<$ on the natural numbers can be described as the smallest transitive relation satisfying $n < n^+$ for all $n \in \mathbb{N}$. From the previous lemma, Lemma 8, we already know that $\in$ is transitive on $\mathbb{N}$.

Lemma 9
The relation $\in$ is the smallest transitive relation r on $\mathbb{N}$ satsifying $\langle n, n^+\rangle \in r$ for all n.

Proof Assume that r is an arbitrary transitive relation on $\mathbb{N}$ satisfying the required condition. We will prove

$$\forall n, m(n \in m \to \langle n, m\rangle \in r)$$

This is best proved by induction on m. For $m = 0$ the statement is vacuously true. So assume $\forall n(n \in m \to \langle n, m\rangle \in r)$ and try to proof $\forall n(n \in m^+ \to \langle n, m^+\rangle \in r)$.

Case $n \in m$ By hypothesis we have $\langle n, m\rangle \in r$. By the stipulated properties of r $\langle m, m^+\rangle$ and transitivity we get $\langle n, m^+\rangle \in r$, as desired.

Case $n = m$ We immediately have $\langle m, m^+\rangle \in r$.

∎

In the set theoretic representation of the natural numbers we chose, we thus discover the curious phenomenon that the $<$-relation cooincides with the $\in$-relation. As a consequence we have that any natural number is the set of all its predecessors, i.e. $n = \{m \mid m < n\}$.

2.3 Recursion

In the last section we have reconstructed in ZF the set $\mathbb{N}$ of natural numbers. But what about addition and multiplication? Of course, we can write down the usual recursive definitions of these functions. The question to be answered is: can we prove existence of these functions from the ZFC axioms? We will first prove a general recursion theorem and later apply it to get addition and multiplication.

Theorem 10 (Recursion Theorem)
Let F be a function satisfying $\mathrm{ran}(F) \subseteq \mathrm{dom}(F)$ *and let u be an element in* $\mathrm{dom}(F)$. *Then there exists exactly one function f satisfying*

1. $\mathrm{dom}(f) = \mathbb{N}$,
2. $f(0) = u$,
3. $f(n^+) = F(f(n))$ *for all* $n \in \mathbb{N}$

Proof The assumptions $\mathrm{ran}(F) \subseteq \mathrm{dom}(F)$ and $u \in \mathrm{dom}(F)$ are needed to make sure that all function applications of F are defined.

Let us first prove the uniqueness part. Thus we start with two functions f and g both satisfying 1-3 from the theorem. Set

$$x = \{y \in \mathbb{N} \mid f(y) = g(y)\}$$

It can be easily seen that $0 \in x$ and for all $n \in \mathbb{N}$ $n \in x$ implies $n^+ \in x$. Thus by the last Peano axiom, the induction axiom, we get $x = \mathbb{N}$, i.e. $f = g$.

To prove existence of f we first notice that there is no ZFC axiom that explicitely relates to recursion. The idea is to instrument the existence of unions, see Lemma 2, for our purposes. Let

$$\begin{aligned} H \quad = \quad \{h \mid \quad & func(h) \wedge h(0) = u \wedge \exists n(n \neq 0 \wedge \mathrm{dom}(h) = n \\ & \wedge \forall m(m^+ \in n \rightarrow h(m^+) = F(h(m))))\} \end{aligned}$$

That is to say H consists of all functions defined on an initial segment of $\mathbb{N}$ (remember that n is the set of all its predecessors) and satisfies the required properties. Now we like to set

$$f = \bigcup H$$

There are some obstacles to be overcome to make this work. First, to appeal to Lemma 2 for the existence of $\bigcup H$ we need to know that H is a set, and not only a class term. Obviously H is a set of ordered pairs with first component from $\mathbb{N}$ and second component from $\mathrm{ran}(F)$, thus $H \subseteq \mathcal{P}(\mathcal{P}(\mathbb{N} \cup \mathrm{ran}(F)))$. The power set axiom and Lemma 2 assure together with the assumptions that $\mathcal{P}(\mathcal{P}(\mathbb{N} \cup \mathrm{ran}(F)))$ is a set. Now we can use the subset axiom A3 to derive that H is indeed a set. So f exists as a set.

Is $\mathrm{dom}(f) = \mathbb{N}$?

The proof procedes by induction. Since for $h_0 = \{\langle 0, u\rangle\}$ we have $h_0 \in H$, we get immediately $0 \in \mathrm{dom}(f)$. If n is in $\mathrm{dom}(f)$ then there is $h \in H$ with $n \in \mathrm{dom}(h)$. Either n^+ is already in $\mathrm{dom}(h)$ or $\mathrm{dom}(h) = n^+$. In the latter case we set $h_+ = h \cup \{\langle n^+, F(h(n))\rangle\}$. Obviously $h_+ \in H$ and therefore $n^+ \in \mathrm{dom}(f)$.

Is f a function?

We prove by induction on x the claim $unique(x)$

$$\forall y_1, y_2((\langle x, y_1\rangle \in f \wedge \langle x, y_2\rangle \in f) \rightarrow y_1 = y_2)$$

Since all $h \in H$ are required to satisfy $h(0) = u$ we have $unique(0)$. Now assume $unique(n)$ and try to infer $unique(n^+)$. Consider y_1, y_2 with $\langle n^+, y_1\rangle \in f$ and $\langle n^+, y_2\rangle \in f$. There are thus $h_1, h_2 \in H$ with $\langle n^+, y_i\rangle \in h_i$. By definition of H we have for both $i \in \{1, 2\}$ $y_i = h_i(n^+) = F(h_i(n))$. Since by induction hypothesis $h_1(n) = h_2(n)$, we also get $y_1 = y_2$.

Does f satisfy $f(0) = u$?

This is obvious.

Is $f(n^+) = F(f(n))$ true for all n?

This is easy. There is $h \in H$ with $h(n^+) = f(n^+)$. By definition of H we have $h(n^+) = F(h(n))$ and since f is a function we must also have $h(n) = f(n)$.

∎

Lemma 11

1. *for every $m \in \mathbb{N}$ there is a unique function add_m such that*

$$\begin{array}{lcl} add_m(0) & = & m \\ add_m(n^+) & = & (add_m(n))^+ \end{array}$$

2. *for every $m \in \mathbb{N}$ there is a unique function $mult_m$ such that*

$$\begin{array}{lcl} mult_m(0) & = & 0 \\ mult_m(n^+) & = & add_m(mult_m(n)) \end{array}$$

Proof

Ad 1 Apply the recursion theorem with $u = m$ and $F(x) = x^+$

Ad 2 Apply the recursion theorem with $u = 0$ and $F_m(x) = add_m(x)$.

∎

Definition 11

1. $x + y =_{def} add_x[y]$
2. $x * y =_{def} mult_x[y]$

Now the usual arithmetical laws for addition and multiplication can be defined. Their proofs do not make reference to set theory any more, all that is needed are the Peano axioms.

2.4 Integers

For this and the following two sections we follow the book [Rubin, 1967]. The purpose of this section is to define the set of all integers $\mathbb{Z}$ using purely set theoretical methods and making use of the already constructed natural numbers $\mathbb{N}$. The basic idea is to represent an integer z by an odered pair $\langle n, m\rangle$ of natural numbers $n, m \in \mathbb{N}$ such that $z = m - n$. The pairs $\langle 7, 5\rangle$ and $\langle 10, 8\rangle$ will then represent the same integer. This leads us to the plan to represent an integer not by a single pair of natural numbers, but by an equivalence class of pairs of natural numbers. Two pairs $\langle m_1, n_1\rangle$ and $\langle m_2, n_2\rangle$ are equivalent if $m_1 - n_1 = m_2 - n_2$. The problem is, that we do not have at the start unrestricted subtraction at our disposal. Fortunately, we can equivalently rephrase the above equation as $m_1 + n_2 = m_2 + n_1$ where subtraction is not involved. This leads us to the following definition.

Definition 12 *For $m, n, p, q \in \mathbb{N}$ we define*

$$\langle m, n\rangle =_i \langle p, q\rangle \Leftrightarrow_{def} m + q = p + n$$

It can be easily verified that $=_i$ is an equivalence relation.

Definition 13

1. *For $m, n \in \mathbb{N}$ we define the equivalence class $[\langle m, n\rangle]_i$ of the pair $\langle m, n\rangle$ with respect to the relation $=_i$, as usual, by:*

$$[\langle m, n\rangle]_i = \{\langle p, q\rangle \mid \langle m, n\rangle =_i \langle p, q\rangle\}$$

2. $\mathbb{Z} = \{u \mid \exists m \exists n(m \in \mathbb{N} \wedge n \in \mathbb{N} \wedge u = [\langle m, n\rangle]_i)\}$

This definition introduces $\mathbb{Z}$ as a class term. But we have

Lemma 12 *$\mathbb{Z}$ is a set.*

Proof We already know that $\mathbb{N}$ is a set. Thus for $m, n \in \mathbb{N}$ we know that $\langle m, n\rangle \in \mathcal{P}^2(\mathbb{N})$, $[\langle m, n\rangle]_i \in \mathcal{P}^3(\mathbb{N})$ and $\mathbb{Z} \subseteq \mathcal{P}^3(\mathbb{N})$. Thus the powerset axiom together with the subset axiom ensures that $\mathbb{Z}$ is indeed a set.

∎

We would like to say when an element $z \in \mathbb{Z}$ is positive or negative. Since z is an equivalence class of pairs of natural numbers the following lemma is needed as a preparation.

Lemma 13 *Let $m, n, p, q \in \mathbb{N}$ and assume $\langle m, n\rangle =_i \langle p, q\rangle$.*
Then the following is true

1. $m < n \leftrightarrow p < q$
2. $m = n \leftrightarrow p = q$
3. $m > n \leftrightarrow p > q$

Proofs

zu 1:

zu 2: We need to show, by induction on m, that $m + q = p + m$ implies $p = q$. For $m = 0$ this follows from $0 + q = q$ and $p + 0 = p$. For $m = 1$ we get $1 + q = q^+ = p + 1 = p^+$ which gives by Peano's axioms $p = q$. Now assume the for all p, and q we know $m + q = p + m$ implies $q = p$ and we aim to show $(m + 1) + q = p + (m + 1)$ implies $q = p$. Making use of associativity and commutative of $+$ we get $m + (q + 1) = (p + 1) + m$. This yields by induction hypothesis $q + 1 = p + 1$ which in turn implies $q = p$ as we have shown before.

zu 3: ∎

On the basis of Lemma 13 we can unambiguously define

Definition 14 *Let $[\langle m, n\rangle]_i \in \mathbb{Z}$.*

1. $[\langle m,n\rangle]_i$ *is negative* $\Leftrightarrow_{def}$ $m < n$
2. $[\langle m,n\rangle]_i$ *is zero* $\Leftrightarrow_{def}$ $m = n$
3. $[\langle m,n\rangle]_i$ *is positive* $\Leftrightarrow_{def}$ $m > n$

The following lemma can easily be verified

Lemma 14 *Let* $[\langle m,n\rangle]_i \in \mathbb{Z}$.

- $[\langle m,n\rangle]_i$ *is negative* $\Leftrightarrow$ *there is a unique* $p \in \mathbb{N}$, $p \neq 0$ *with* $\langle m,n\rangle =_i \langle 0,p\rangle$
- $[\langle m,n\rangle]_i$ *is zero* $\Leftrightarrow$ $\langle m,n\rangle =_i \langle 0,0\rangle$
- $[\langle m,n\rangle]_i$ *is positive* $\Leftrightarrow$ *there is a unique* $p \in \mathbb{N}$, $p \neq 0$ *with* $\langle m,n\rangle =_i \langle p,0\rangle$

The arithmetic operations can be defined on $\mathbb{Z}$ as follows

Definition 15 *Let* $[\langle m,n\rangle]_i, [\langle p,q\rangle]_i \in \mathbb{Z}$, *then*

- $[\langle m,n\rangle]_i + [\langle p,q\rangle]_i =_{def} [\langle m+p, n+q\rangle]_i$
- $[\langle m,n\rangle]_i - [\langle p,q\rangle]_i =_{def} [\langle m+q, n+p\rangle]_i$
- $[\langle m,n\rangle]_i * [\langle p,q\rangle]_i =_{def} [\langle mp+nq, mq+np\rangle]_i$

For these definitions to make sense we have to show that they are independent of the choice of representatives of the involved equivalence classes, i.e. we have to prove the following lemma

Lemma 15 *Let* $m_1, m_2, n_1, n_2, p_1, p_2, q_1, q_2 \in \mathbb{N}$ *be given with* $\langle m_1,n_1\rangle =_i \langle m_2,n_2\rangle$ *and* $\langle p_1,q_1\rangle =_i \langle p_2,q_2\rangle$. *Then*

1. $\langle m_1+p_1, n_1+q_1\rangle =_i \langle m_2+p_2, n_2+q_2\rangle$
2. $\langle m_1+q_1, n_1+p_1\rangle =_i \langle m_2+q_2, n_2+p_2\rangle$
3. $\langle m_1p_1+n_1q_1, m_1q_1+n_1p_1\rangle =_i \langle m_2p_2+n_2q_2, m_2q_2+n_2p_2\rangle$

Proof Easy computation.

As a last item we define the order relations on $\mathbb{Z}$:

Definition 16 *Let* $[\langle m,n\rangle]_i, [\langle p,q\rangle]_i \in \mathbb{Z}$*, then*

- $[\langle m,n\rangle]_i < [\langle p,q\rangle]_i \Leftrightarrow_{def} [\langle m,n\rangle]_i - [\langle p,q\rangle]_i$ *is negative*
- $[\langle m,n\rangle]_i \leq [\langle p,q\rangle]_i \Leftrightarrow_{def} [\langle m,n\rangle]_i < [\langle p,q\rangle]_i$ *or* $[\langle m,n\rangle]_i = [\langle p,q\rangle]_i$

As an easy consequence of Definitions 14 and 15 we get

Lemma 16 *For* $[\langle m,n\rangle]_i, [\langle p,q\rangle]_i \in \mathbb{Z}$

1. $[\langle m,n\rangle]_i < [\langle p,q\rangle]_i \leftrightarrow m+q < n+p$
2. $[\langle m,n\rangle]_i \leq [\langle p,q\rangle]_i \leftrightarrow m+q \leq n+p$

In the same way one can continue to define the rational and real numbers as sets and derive their known properties from the ZF axioms. Details may be found in [Rubin, 1967, Chapter 6].

2.5 Ordinals

The theory of transfinite ordinal numbers is one of the most fascinating areas of set theory. The *Compact Oxford English Dictionary* explains the term

> *ordinal number* or just *ordinal* as a noun designating a number defining a thing's position in a series, such as 'first' or 'second'.

In the context of axiomatic set theory, which is devoted to the study of infinite sets, it is tempting to try to also extend counting into the infinite. The convey a first idea how this could be done considering first the natural numbers in their natural ordering. The number *6* occurs, not surprisingly at the sixth position. Let us look at another sequencing

$$\begin{array}{lll} n < m \Leftrightarrow & n = 2*n_0 \wedge m = 2*m_0 \wedge n_0 < m_0 & \textit{or} \\ & n = 2*n_0 \wedge m = 2*m_0 + 1 & \textit{or} \\ & n = 2*n_0 + 1 \wedge m = 2*m_0 + 1 \wedge n_0 < m_0 & \end{array}$$

In this ordering first come the even numbers in their usual ordering and after all of them come the uneven numbers again in their usual sequence:

$$0, 2, 4, 6, \ldots 1, 3, 5, \ldots$$

Now number *6* occurs at the forth position. To describe the positions of the uneven numbers we need an extension of the finite ordinal numbers. These will be called the *transfinite* ordinals. The number *1* will occur at the first infinite positision, tradionally denoted by ω. Number *5* will occur at position $\omega + 2$. That this simple idea can be turned into a consistent mathematical theory is the content of the remainder of this section.

As a first observation we note that not any arbitrary linear ordering is suitable for counting, e.g., with the integers in their usual ordering there would be no first number to start with. We need to require that after we have counted a subset X of an ordered set $(G, <)$ there is a next element after X, i.e., a least element that is strictly greater than all $g \in X$. Here is a formal definition,

Definition 17 (Well-founded Ordering) *A linearly ordered set $(G, <)$ is called* well-founded *if for any subset $X \subset G$ such that there is g with $\forall x(x \in X \to x < g)$ there is a least such $g_0 \in G$, i.e.*

1. $\forall x(x \in X \to x < g_0)$
2. $\forall h(\forall x(x \in X \to x < h) \to g_0 \leq h)$

We call g the next element *after X.*
Quite frequently the term wellorder *is used instead of well-founded ordering.*

Most of the time the property of the following lemma is used as a definition instead of Definition 17.

Lemma 17 *A linear order $(G, <)$ is well-founded if and only if every non-empty subset $Y \subseteq G$ has a least element.*

Proof Assume first that $(G, <)$ satisfies the property from Definition 17 and consider $\emptyset \neq Y \subseteq G$. We need to find a least element of Y. Set $Z = \{g \in G \mid \forall h(h \in Y \to g < h)\}$. Since $Y \neq \emptyset$ there is at least one $g_1 \in G$ striclty greater than all elements of Z. By assumption there is thus a next

element g after Z. It is easy to see that g is the least element of Y.
For the reverse implication we need to find a next element after some $X \subset G$ that does not dominate every element in G. Set $W = \{g \in G \mid \forall h(h \in X \rightarrow h < g)\}$. By choice of X we know $W \neq \emptyset$ and there is thus a least element g_0 of W. Again, it is easy to see that g_0 is the next element after X.

▪

Here is another characterisation of wellorders

Lemma 18 *A linear order $(G, <_1)$ is well-founded if there is no function $f : \mathbb{N} \rightarrow G$ such that for all $n, m \in \mathbb{N}$ $n < m$ implies $f(m) <_1 f(n)$.*
One usually summarizes this condition by saying that there is no infinite descending chain f in $(G, <_1)$.

We needed two symbols $<$ and $<_1$ to distinguish the ordering of the natural numbers from the ordering on G.

Proof Assume there is an infinite descending chain f in $(G, <_1)$ then $Y = range(f)$ contradicts Lemma 17.
If on the other hand $Y \subseteq G$ is a non-empty subset $Y \subseteq G$ without a least element then define $f : \mathbb{N} \rightarrow G$ by

$$\begin{array}{lcl} f(0) & = & \text{an arbitrary element } g_0 \text{ in the non-empty set } Y \\ f(n+1) & = & \text{an element } g \in Y \text{ with } g <_1 f(n) \end{array}$$

Obviously f is an infinite descending chain.

▪

Lemma 19 (Principle of Well-founded Induction)
Let $(G, <)$ be well-founded relation, ϕ a first-order formula such that

$$\forall x \in G((\forall y(y \in G \wedge y < x \rightarrow \phi(y)) \rightarrow \phi(x))$$

then

$$\forall x \in G(\phi(x))$$

Note, that the assumption of the lemma implies as a special case $\phi(g_0)$ for the least element of $(G, <)$.

Proof Set $H = \{x \in G \mid \phi(x)\}$. Assume $G \neq H$. Then $G \setminus H$ is non-empty and thus there is a least element $g_1 \in G \setminus H$ by Lemma 17. By minimality of g_1 we get $\phi(x)$ for every $x < g_1$. Our assumption entails also $\phi(g_1)$. This contradicts $g_1 \notin H$. Thus we must have $G = H$.

∎

Lemma 20 (Alternative Principle of Well-founded Induction)
Let $(G, <)$ be well-founded relation, a a set such that

$$a \neq \emptyset \wedge \forall x \in a \ (\forall y(y \in G \wedge y < x \rightarrow y \in a) \rightarrow x \in a)$$

then

$$u = G$$

Proof Use the Lemma 19 with $\phi(x) = x \in a$.

∎

We will later need the following easy construction.

Definition 18 (Initial Segment)
Let $(G, <)$ be an ordered set and $a \in G$ such that a is not the least element in $(G, <)$.
The structuture $(G_a, <_a)$ is defined by

1. $G_a = \{g \in G \mid g < a\}$
2. $g_1 <_a g_2 \Leftrightarrow g_1 < g_2$ *for* $g_1, g_2 < a$.

$(G, <)$ is called a initial segment *of $(G, <)$*

We note the following easy facts.

Lemma 21 *Let $(G, <)$ be an ordered set and $a \in G$ such that a is not the least element in $(G, <)$.*

1. *$(G_a, <_a)$ is again an ordered set.*
2. *If $(G, <)$ is a well-ordered set then $(G_a, <_a)$ is also a well-ordered set.*

Proof Easy.

▪

The previous Lemma 19 can be generalized to the case $(G, <)$ where G is only described by a class term and need not be a set. If G is replaced by a class term also $<$ has to be replaced by a formula, called WO in the next lemma. It suffices to consider WO, G could be recovered as the domain of WO if needed.

Lemma 22 (Extended Principle of Well-founded Induction)
Let $WO(x, y)$ be a formula with the two free variables x, y such that

1. $ZF \vdash \forall u(u \neq \emptyset \rightarrow \exists v(v \in u \wedge \forall z(z \in u \rightarrow \neg WO(z, v))))$
   every non-empty set contains a WO-minimal element
2. $ZF \vdash \forall x \exists u(x \subseteq u \wedge \forall z, v(z \in u \wedge WO(v, z) \rightarrow v \in u))$
   every set can be extended to a superset that is closed under predecessors of WO.

For any formula ϕ with $ZF \vdash \forall x(\forall y(WO(y, x) \rightarrow \phi(y)) \rightarrow \phi(x))$ we obtain

$$ZF \vdash \forall x \phi(x)$$

Proof This proof parallels the proof of Lemma 19. If $\forall x \phi(x)$ is not true there is a set g with $\neg\phi(g)$. Let u be a superset of $\{g\}$ closed under WO predecessors which exists by assumption. Define $u' = \{z \in u \mid \neg\phi(z)\}$. Since $g \in u$, we have $u' \neq \emptyset$. By the properties of WO u' contains a WO minimnal element g_1, i.e., $g_1 \in u'$ and forall h with $WO(h, g_1)$ we get $h \notin u'$. By choice of u we have $h \in u$. Thus we obtain $\phi(h)$ forall h with $WO(h, g_1)$. By the assumed property of ϕ this implies $\phi(g_1)$ a contradiction.

▪

Example 1

1. *The most simple example of a wellorder is* $(\mathbb{N}, <)$.

2. *The structure* $(\mathbb{N}, <_1)$ *with*

$$n <_1 m \Leftrightarrow \begin{cases} n < m & \text{and both } n \text{ and } m \text{ are even} \\ n < m & \text{and both } n \text{ and } m \text{ are uneven} \\ n \text{ is even and } m \text{ is uneven} \end{cases}$$

can also easily be seen to be a wellorder.

3. *We generalize the idea from example 2. For a number* n *let* $minp(n)$ *be the smallest prime divisor for* $n \in \mathbb{N}$ *and* $n \notin \{0, 1\}$. *We now define* $(\mathbb{N}, <_2)$

$$n <_2 m \Leftrightarrow \begin{cases} n = 0 \text{ and } m \neq 0 & \\ n = 1 \text{ and } 1 < m & \\ 1 < n < m & \text{and } minp(n) = minp(m) \\ 1 < n, m & \text{and } minp(n) < minp(m) \end{cases}$$

Using Lemma 18 we can see that $(\mathbb{N}, <_2)$ *is a wellorder as follow. Assume there is an infinite descending chain*

$$n_0 >_2 n_1 \ldots >_2 n_k >_2 \ldots$$

Then we must have from some point r *on that for all* $k > r$ $minp(n_k) = minp(n_r)$. *By definition of* $>_2$ *this would yield an infinite descending chain* $n_r > n_{r-1} \ldots > n_k > \ldots$ *in* $\mathbb{N}, <)$. *A contradiction!*

∎

Our goal can now be stated more precisely: to find numbers for counting all well-founded orderings. We start with the following definition

Definition 19 (Ordinal Numbers) *A set* a *is called an* ordinal *if it is transitive and satisfies*
$\forall x \forall y (x \in a \wedge y \in a \rightarrow (x \in y \vee y \in x \vee x = y))$.

Lemma 23

1. 0 *is an ordinal*

2. *If* α *is an ordinal then* α^+ *is also an ordinal.*
   Thus every natural number is an ordinal.

3. *The set of all natural numbers, traditionally denoted by the letter ω, is an ordinal.*

4. *If α is an ordinal, then every element $\beta \in \alpha$ is an ordinal.*

Proofs
ad 1 trivial
ad 2 See Exercise 2.11.2 .
ad 3 Follows from the fact that $\in$ is the (linear) order relation on the natural numbers and Lemma 8(2) .
ad 4 By transitivity we know $\beta \subseteq \alpha$, thus the $\in$-relation restricted to β is a linear order, since it is already a linear order on the larger set α. It remains to show transitivity. So assume $w \in v$ and $v \in \beta$ with the aim to show $w \in \beta$. From $v \in \beta \subseteq \alpha$ we get $v \in \alpha$, thus also $w \in v \subseteq \alpha$ and $w \in \alpha$. Since the $\in$-relation is a linear order on α we must have $\beta \in w$ or $\beta = w$ or $w \in \beta$. Since the first two alternatives contradict the foundation axiom we get $w \in \beta$, as needed.

Lemma 24

1. *Let α be an ordinal and x a transitive set, then*

$$x \subset \alpha \leftrightarrow x \in \alpha$$

2. *If α and β are ordinals then $\alpha \cap \beta$ is also an ordinal.*

3. *For ordinals α and β we have*

$$\alpha \in \beta \vee \alpha = \beta \vee \beta \in \alpha$$

4. *If b is a set of ordinals then $\bigcup b$ is also an ordinal.*

Proofs
ad 1 The implication from right to left is rather simple. Since α is transitive, we immediately get $x \subseteq \alpha$. Since x itself cannot be an element of x we also get strict inclusion $x \subset \alpha$.
The reverse implication is a little bit more demanding. By simple Boolean

algebra we get from $x \subset \alpha$ that $\alpha \setminus x \neq \emptyset$. By the foundation axioms (see also Exercise 2.11.1) there is set $y \in (\alpha \setminus x)$ satisfying $(\alpha \setminus x) \cap y = \emptyset$. Since α is in particular transitive we have $y \subseteq \alpha$ and further $y \subseteq x$. If we can show also $x \subseteq y$, then we arrive at $x \in \alpha$ and are finished.
So let us consider $z \in x$. Since $x \subseteq \alpha$ we also get $z \in \alpha$. By the defining property of ordinal we get $y \in z \vee y = z \vee z \in y$. The first two alternatives imply, using for the first time the transitivity assumption on x that $y \in x$, which contradicts the choice of y. Thus we must have $z \in y$.
ad 2 Straightforward.
ad 3 We start with the obvious observations $\alpha \cap \beta \subseteq \alpha$ and $\alpha \cap \beta \subseteq \beta$. We claim that at least one of these set inclusions has to be an equality. Otherwise we would have $\alpha \cap \beta \subset \alpha$ and $\alpha \cap \beta \subset \beta$ which would yield by part one of the lemma $(\alpha \cap \beta) \in \alpha$ and $(\alpha \cap \beta) \in \beta$. But this leads to the contradiction $(\alpha \cap \beta) \in (\alpha \cap \beta)$. $(\alpha \cap \beta) = \alpha$ would imply by the first part of this lemma $\alpha = \beta \vee \alpha \in \beta$, while $(\alpha \cap \beta) = \beta$ would imply $\alpha = \beta \vee \beta \in \alpha$.
ad 4 Let $c = \bigcup b$. We first show that c is transitive. For $x \in c$ there is by definition of the sum operation an element $y \in b$ with $x \in y$. Since y is by assumption transitive we get $x \subseteq y$ and since $y \subseteq c$ we also have $x \subseteq c$.

If $x, y \in c$ then in particular x, y are ordinals and we get $x \in y \vee x = y \vee y \in x$ from part 3 of this lemma.

∎

Definition 20

1. *An ordinal α such that $\alpha = \beta^+ = \beta \cup \{\beta\}$ for some β is called a* successor ordinal.

2. *An ordinal α such that for all β with $\beta \in \alpha$ there is γ such that $\beta \in \gamma$ and $\gamma \in \alpha$ is called a* limit ordinal.

Lemma 25

1. *For every ordinal α the structure (α, ϵ) is a wellorder.*

2. *$(\alpha, \epsilon) \cong (\beta, \epsilon)$ implies $\alpha = \beta$.*

Proof
1 To show linearity consider $\beta_1, \beta_2, \beta_3 \in \alpha$ with $\beta_1 \in \beta_2$ and $\beta_2 \in \beta_3$. By Lemma 23(4) β_3 is an ordinal. Thus $\beta_2 \in \beta_3$ implies $\beta_2 \subseteq \beta_3$. Now, $\beta_1 \in \beta_2$ implies $\beta_1 \in \beta_3$ as desired. Obviously, (α, ϵ) is anti-reflexiv. The linearity axiom is Lemma 24(3).
2 Is left to the reader, see Exercise 2.11.15.

∎

Theorem 26 *For every well-ordered set $(G, <)$ there is a unique ordinal α such that*

$$(G, <) \cong (\alpha, \epsilon)$$

Proof We define

$$u = \{a \in G \mid (G_a, <_a) \cong (\beta, \epsilon) \text{ for some ordinal } \beta\}$$

If a_0 is the least element of $(G, <)$ then trivially $(G_a, <_a) \cong (0, \epsilon)$. Thus $a_0 \in u$ and u is not the empty set.

Before going on we note that if $(G_a, <_a) \cong (\beta, \epsilon)$ is true there can be only one isomorphism $f : (G_a, <_a) \to (\beta, \epsilon)$. If $g : (G_a, <_a) \to (\beta, \epsilon)$ is another isomorphism we proof by well-founded induction that $f(x) = g(x)$ for all $x \in G_a$ as follows. Consider $b \in G_a$, i.e., $b \in G$ and $b < a$ such that for all $c \in G_a$ with $c < b$ we know $f(c) = g(c)$. To apply Lemma 19 or 20 we need to show $f(b) = g(b)$. Obviously, b is the least upper bound of $\{c \in G_a \mid c < b\}$. Since (β, ϵ) is a well-ordering the least upper bound γ of $f(c) \mid c < b\} = g(c) \mid c < b\}$ exists and since f and g are order-isomorphisms we must have $f(b) = \gamma = g(b)$. The argument at the same time also shows that $(G_a, <_a) \cong (\beta, \epsilon)$ and $(G_a, <_a) \cong (\gamma, \epsilon)$ for ordinals β, γ imply $\alpha = \gamma$.

The next step in the proof of the lemma is to show $u = G$. For this we employ again well-founded induction. For $a \in G$ with $b \in u$ for all $b < a$ we have to show $a \in u$. Thus there are for every $b < a$ an ordinal β_b and an isomorphism $f_b : (G_b, <_b) \cong (\beta_b, \epsilon)$. For $b_1 < b_2 < a$ we just observed that $\beta_{b_1} \subset \beta_{b_2}$ and f_{b_2} an extension of f_{b_1}. We denote by c^+ the least element in $(G, <)$ that is strictly greater c. We may thus define unambiguously f on G_a

- $f(b) = f_b(c)^+$ if $b = c^+$ for some $c \in G$ or

- $f(b) =$ the least (strict) upper bound of $\{f_{c^+}(c) \mid c < b\}$ if b is a limit point in $(G, <)$, i.e., for all $c < b$ also $c^+ < b$.

Obviously, f is an order isomorphism from $(G_a, <_a)$ onto $(\bigcup_{b<a} \beta_b, \epsilon)$. By Lemma 24 (4) $\bigcup_{b<a} \beta_b = \alpha$ is an ordinal. This complete the proof of $u = G$.

Piecing together the isomorphisms $(G_a, <_a) \cong (\beta_a, \epsilon)$ to one isomorphis $(G, <) \cong (\alpha, \epsilon)$ works the same way we have put together the isomorphisms $(G_b, <_b) \cong (\beta_b, \epsilon)$ for $b < a$ to an isomorphism $(G_a, <_a) \cong (\beta_a, \epsilon)$.

∎

Example 2 *We reconsider the example of well-orders given in Example 1.*

1. $(\mathbb{N}, <) \cong (\omega, \epsilon)$
2. $(\mathbb{N}, <_1) \cong (\omega + \omega, \epsilon)$
3. $(\mathbb{N}, <_2) \cong (\omega * \omega, \epsilon)$

A1 Extensionality $\forall z(z \in x \leftrightarrow z \in y) \rightarrow x = y.$

A2 Foundation $\exists y(y \in x) \rightarrow \exists y(y \in x \wedge \forall z \neg(z \in x \wedge z \in y)).$

A3 Subset

$$\exists y \forall z(z \in y \leftrightarrow z \in x \wedge \phi(z)).$$

for any formula ϕ not containing y.

A4 Empty set $\exists y \forall x(x \notin y).$

A5 Pair set $\exists y \forall x(x \in y \leftrightarrow x = z_1 \vee x = z_2).$

A6 Power set $\exists y \forall z(z \in y \leftrightarrow \forall u(u \in z \rightarrow u \in x)).$

A7 Sum $\exists y \forall z(z \in y \leftrightarrow \exists u(z \in u \wedge u \in x))$

A8 Infinity

$$\begin{array}{ll} \exists w(\emptyset \in w \wedge & \forall x(x \in w \rightarrow \\ & \exists z(z \in w \wedge \\ & \forall u(u \in z \leftrightarrow u \in x \vee u = x)))) \end{array}$$

A9 Replacement

$$\begin{array}{l} \forall x, y, z(\psi(x, y) \wedge \psi(x, z) \rightarrow y = z) \rightarrow \\ \exists u \forall w_1(w_1 \in u \leftrightarrow \exists w_2(w_2 \in a \wedge \psi(w_2, w_1))) \end{array}$$

A10 Axiom of Choice

$$\begin{array}{l} \forall x(x \in z \rightarrow x \neq \emptyset \wedge \\ \forall y(y \in z \rightarrow x \cap y = \emptyset \vee x = y)) \\ \rightarrow \\ \exists u \forall x \exists v(x \in z \rightarrow u \cap x = \{v\}) \end{array}$$

Figure 2.1: Axioms of Zermelo-Fraenkel Set Theory

2.6 König's Lemma (Optional)

Before we can state and prove König's Lemma we need a couple of definitions that are also interesting in their own right.

Following general practice we will write ω insted of $\mathbb{N}$ in the present context.

Definition 21 (Finite Sets)

1. *A set s is called* finite *if there is a bijective (i.e., injective and surjective) function f from a natural number n onto s.*
   Using the terminoly of Definition 6 we may thus write

$$\begin{aligned} fin(s) \quad \leftrightarrow \quad & \exists n \in \omega \; \exists f(funct(f,n,s) \wedge \\ & \forall x_1 \forall x_2 \forall y((\langle x_1, y\rangle \in f \wedge \langle x_2, y\rangle \in f) \rightarrow x_1 \doteq x_2) \wedge \\ & \forall y(y \in s \rightarrow \exists m \in n(\langle m, y\rangle \in f))) \end{aligned}$$

   Remember, that a natural number n equals the set its predecessors, see remark after Lemma 9.

2. *If s is a finite set with $funct(f,n,s)$ for a bijection f then n is called the* cardinality *of s, in symbols $card(s) = n$.*

3. *A set s is called infinite if it is not finite, i.e., if $\neg fin(s)$.*

Definition 22 (Sequences)

1. *A finite sequence of elements from a set a is a function f from a natural number n into a, i.e. $func(f,n,a)$,*

2. *For a finite sequence f with $func(f,n,a)$ we denote by $len(f) = n$ the length of f. The empty set $\emptyset$ thus is a sequence of length 0.*

3. *An infinite sequence of elements from a set a is a function f from ω into a, i.e. $func(f,\omega,a)$,*

4. *If f is a sequence, finite or infinite, of elements from a, i.e., $funct(f,\alpha,a)$ for $\alpha = \omega$ or $\alpha \in \omega$ and $m \in \alpha$ then $f \restriction m$ is the restriction of f to m,*

$$f \restriction m = \{\langle n, y\rangle \mid \langle n, y\rangle \in f \wedge n < m\}$$

5. *If f is a sequence of elements from a of finite length n, i.e., $funct(f, n, a)$ and $m \leq n$ then $tail(f, m)$ is the tail of f starting at m,*

$$tail(f, m) = \{\langle k, x\rangle \mid \langle k + m, x\rangle \in f\}$$

Informally we may write $tail(x_0, \ldots x_m, \ldots x_{n-1}, m) = x_m, \ldots x_{n-1}$. In particular, $tail(f, len(f))$ is the empty sequence.

6. *If f is a finite sequence of elements from a of length k, $func(f, k, a)$, and $x \in a$ then $f^\frown x$ is the sequence*

$$f^\frown x = f \cup \{\langle k, x\rangle\}$$

of length $k + 1$.

Definition 23 (Trees)

1. *A tree T of elements from a set a is a set of finite sequences of elements from a closed under restriction, i.e.,*

   (a) $\forall t(t \in T \rightarrow \exists n(n \in \omega \wedge func(t, n, a)$ *and*

   (b) $\forall t \forall n(t \in T \wedge n < len(t) \rightarrow (t \restriction n) \in T)$

   Think of the nodes of a tree T as the end points of its finite sequences. The empty sequence thus represents the root of the tree and every $f^\frown x$ is a successor of f.

2. *For a tree T and a node $t \in T$ with $func(t, n, a)$ we denote by $T \restriction t$ the subtree of T rooted in t, i.e.,*

$$T \restriction t = \{tail(g, n) \mid g \in T \wedge (g \restriction n) = t\}$$

3. *For a tree T and a node $t \in T$ with $func(t, n, a)$ we denote by $succ(T, t)$ the set of successors of t in T,*

$$succ(T, t) = \{t^\frown x \mid x \in a \wedge (t^\frown x) \in T\}$$

4. *A tree T is called finitely branching if for all $t \in T$ the sets $succ(T, t)$ are finite. Note, there need not be an upper bound on the cardinality of the successor sets.*

5. *A branch p of a tree T of elements from a is an infinite sequence of elements from a, i.e., $func(p, \omega, a)$ such that for all $n \in \omega$ $(f \restriction n) \in T$.*

We will need the following simple observation

Lemma 27
Let T be a tree of elements from a and $t \in T$.
Then $T \restriction t$ is the disjoint union of $\{\emptyset\}$ and the sets $T \restriction (t^\frown x)$ for $x \in a$ and $t^\frown x \in T$

Proof Easy.

∎

Lemma 28 (König's Lemma)
Let T be an infinite, finitely branching tree.
Then there is an infinite branch in T.

Proof This will be the first application of the axiom of choice $A10$:

$$\begin{array}{l} \forall x(x \in z \rightarrow x \neq \emptyset \wedge \\ \forall y(y \in z \rightarrow x \cap y = \emptyset \vee x = y)) \\ \rightarrow \\ \exists u \forall x \exists v(x \in z \rightarrow u \cap x = \{v\}) \end{array}$$

in these lecture notes.

Consider the following two set

$$IS(T,t) = \{t^\frown x \in T \mid T \restriction (t^\frown x) \text{ is infinite}\} \tag{2.1}$$

$$Z = \{IS(T,t) \mid t \in T \text{ such that } T \restriction t \text{ is infinite}\} \tag{2.2}$$

We claim that every set $IS(T,t) \in Z$ is not empty. So consider $t \in T$ with $T \restriction t$ infinite. We argue that $IS(T,t) \neq \emptyset$. Since T is assumed to be finitely branching we get from Lemma 27 that $T \restriction t$ is the finite union of $\{\emptyset\}$ and $T \restriction (t^\frown x_i)$ with $i < k$ for some k. By exercise 2.11.17 at least one of the $T \restriction (t^\frown x_i)$ will be infinite. Thus $t^\frown x_i \in IS(T,t)$.

Furthermore, if $IS(T,t_1) \in Z$ and $IS(T,t_2) \in Z$ for $t_1 \neq t_2$ then $IS(T,t_1) \cap IS(T,t_2) = \emptyset$. If t_1 and t_2 are of different length then also all element in $t \in IS(T,t_1)$ have a length different from all elements in $IS(T,t_2)$. In case $len(t_1) = len(t_2) = n$ the existence of $t \in IS(T,t_1) \cap IS(T,t_2)$ would yield the contradiction $t_1 = t \restriction n) = t_2$. The axiom of choice (instantiate the parameter z with the set Z) yields the existence of a set U with $U \cap IS(T,t)$ a singleton set for all $IS(T,t) \in Z$. Let ch be the function from Z to T with $U \cap IS(T,t) = \{ch(IS(T,t)\}$. An infinite branch p of T can now be defined by recursion:

$$\begin{array}{lcl} p(0) & = & \emptyset \\ p(n+1) & = & ch(IS(T,p(n))) \end{array}$$

To show that this is a good definition we need to show that for all n the argument $IS(T,p(n))$ is in the domain Z of the function ch. Be definition $IS(T,p(n)) \in Z$ if $T \restriction p(n)$ is infinite. We therefore show be induction on n the $T \restriction p(n)$ is infinite. For $n = 0$ we get infinity of $T \restriction \emptyset = T$ from the assumption of the lemma. The inductive step follows from the definition of $IS(T,t)$.

∎

2.7 Cardinals (Optional)

Cardinals or cardinal numbers are used to *count* the size of a set. It is one of the great achievments of set theory to have extended counting to infinite sets. Even more fundamental than the cardinal numbers themselves are the definitions when two sets are said to contain the same number of elements, or when one set contains strictly less elements than another one.

Definition 24 (Equivalence)

1. *Two sets a, b are called equivalent, in symbols $a \simeq b$, if there is a injective and surjective function $f : a \to b$.*
   In German equivalent sets are called gleichmächtig.
2. *A set a is smaller than b, in symbols $a \precsim b$ if there is an injective function $f : a \to b$.*

Example 3

1. *For $n, m \in \mathbb{N}$* $\quad n \simeq m \Leftrightarrow n = m$
2. $\mathbb{N} \simeq \mathbb{Z}$
   Here is an isomorphism $F : \mathbb{N} \to \mathbb{Z}$
   $$\begin{array}{lcll} f(0) & = & 0 & \\ f(2n) & = & n & \textit{for} n \neq 0 \\ f(2n+) & = & -n & \textit{for} n \geq 0 \end{array}$$
3. $\mathbb{N} \simeq \mathbb{Q}$ $\quad \mathbb{Q}$ *the set of rationals*

Lemma 29 (Basic Properties)

1. *$\simeq$ is an equivalence relation.*
2. *$\precsim$ is a reflexive order relation.*
3. *$a \precsim b$ and $b \precsim a$ implies $a \simeq b$.*
4. *For any two sets a,b we have*
   $$a \prec b \text{ or } b \prec a \text{ or } a \simeq b.$$

Proof Items (1) and (2) are easy. Item (3) is quite intricate and goes by the name of *Schröder-Bernstein-Theorem*. Item (4) finally uses the axiom of choice (and is in fact equivalent to it). For all proofs see e.g. [Takeuti & Zaring, 1971, Section 10].

∎

Lemma 30 (Cardinality of the Powerset) *For all sets a*

$$a \prec \mathcal{P}(a)$$

Proof The function $sg : a \to \mathcal{P}(a)$ that sends an element $x \in a$ to its singleton set $\{x\} \in \mathcal{P}(a)$ shows $a \precsim \mathcal{P}(a)$. If $f : a \to \mathcal{P}(a)$ is a bijection we define the subset $c \subseteq a$ by

$$x \in c \Leftrightarrow x \notin f(x)$$

Since f is a bijection there will be some $y \in a$ with $c = f(y)$. But, this yields the contradiction $y \notin f(y) \Leftrightarrow y \in c$.

∎

2.8 Ramsey Theory(Optional)

This section presents statements and proofs for a selction of results from what is now called Ramsey Theory. This theory can be divided into a part dealing with infinitary combinatorics and another addressing finite combinatorics.

2.8.1 Infinite Ramsey Theory

The material presented in this section is adapted from [Kunen, 1977].

We first establish the most important definition in this section.

Definition 25 *Let I be an arbitrary set and $n \in \omega$. By $[I]^n$ we denote the set of n-element subsets of I*

$$[I]^n = \{t \mid t \subseteq I \wedge card(t) = n\}$$

In particular we get $[N]^2 = \{\{x, y\} \mid x, y \in N \wedge x \neq y\}$

Definition 26 *1. We call Q a partition of a set X, in symbols $Part(Q, X)$, if it is a set of mutually disjoint subsets of X whose union gives all of X:*

$$\forall y(y \in Q \rightarrow y \subseteq X)$$
$$\forall y_1 \forall y_2(y_1 \in Q \wedge y_2 \in Q \rightarrow y_1 = y_2 \vee y_1 \cap y_2 = \emptyset)$$
$$\bigcup Q = X$$

2. A finite partition $Q = \{Q_0, \ldots, Q_{k-1}\}$ of X, $Part(Q, X)$, may be equivalently represented as a function P from X to k via

$$P(x) = i \Leftrightarrow x \in Q_i$$

A partition G of $[N]^2$ into two parts, i.e., $G : [N]^2 \rightarrow 2$, can be identified with an undirected graph without self-loops on the node set N. Two different nodes $g, h \in N$ are connected if $G(\{x, y\}) = 1$.

Definition 27 *Let $P : [I]^n \rightarrow k$ be a partition of a set $[I]^n$ into k pieces. A subset $H \subseteq I$ is called homogeneous for P if there is $n < k$ such that $P(t) = k$ for all $t \in [H]^n$, i.e., $[H]^n$ is contained in one part of the partition P.*

Theorem 31 (Ramsey's Theorem)
For every $n, k \in \omega$ every partition $P : [\omega]^n \to k$ has an infinite homogeneous set.

The statement of this theorem is usually abbreviated as $\omega \to (\omega)^n_k$.
The special case $\omega \to (\omega)^2_2$ says that every infinite undirected graph contains an infinite subgraph that is either empty (no two nodes are connected) or complete (any two different nodes are connected).

The rest of this section will be devoted to a proof of Theorem 31. The proof proceeds by induction on n. For $n = 1$ the claim of Theorem 31 is reduced to

> If an infinite set is partioned into finitely many disjoint subsets, at least one of the subsets has to be infinite.

This is intuitively obvious. See Exercise 2.11.17 and its solution for a proof from the ZF axioms.

Definition 28 *Let $P : [\omega]^{n+1} \to k$ be a partition. A subset $H \subseteq \omega$ is called pre-homogeneous for P if P on $[H]^{n+1}$ does not depend on the last element of an $(n+1)$-tuple, precisely:*

$$\text{For all } x_0, \ldots, x_{n-1}, y, z \in H \text{ with } x_0 < \ldots x_{n-1} < y \text{ and } x_0 < \ldots x_{n-1} < z$$
$$P(\{x_0, \ldots, x_{n-1}, y\}) \quad = \quad P(\{x_0, \ldots, x_{n-1}, z\})$$

For $\bar{x} = \{x_0, \ldots, x_{n-1}\} \in [\omega]^n$ and $y \in \omega$ we will write $\bar{x} < y$ to abbreviate $\bigwedge_{i=0}^{i=n-1} x_i < y$. The pre-homogeneity property of H for P can thus be formulated as:
For all $\bar{x} \in [H]^n$ and $y, z \in H$ with $\bar{x} < y$ and $\bar{x} < z$ we have
$$P(\bar{x} \cup \{y\}) = P(\bar{x} \cup \{z\}).$$

Lemma 32 *For every partition $P : [\omega]^{n+1} \to k$ with $k \in \omega$ there is an infinite pre-homogenous subset H for P.*

Proof of Theorem 31 using Lemma 32 To prove the inductive step we start with a partition $P : [\omega]^{n+1} \to k$. By Lemma 32 there is an infinite pre-homogeous set $H \subseteq \omega$ for P. We define a partition Q on $[H]^n$ by

$$Q(\bar{h}) = k \quad \Leftrightarrow \quad \exists h \in H(\bar{h} < h \ \wedge P(\bar{h} \cup \{h\}) = k)$$

Since H is pre-homogeneous for P there cannot be h_1, h_2 with $\bar{h} < h_1$, $\bar{h} < h_2$ and $P(\bar{h} \cup \{h_1\}) \neq P(\bar{h} \cup \{h_2\})$. Thus, the definition for Q is sound.

By induction hypothesis there is an infinite homogeneous subset $K \subseteq H$ for Q, with, say, $Q(\bar{k}) = j$ for some $0 \leq j < k$ and all $\bar{k} \in [K]^n$. Since H was pre-homogeneous we have:

$$Q(\bar{k}) = j \quad \Leftrightarrow \quad \forall u \in K(\bar{k} < u \to P(\bar{k} \cup \{u\}) = j)$$

Since any $\bar{k} \in [K]^{n+1}$ may be written as $\bar{k_0} \cup \{u\}$ with $\bar{k_0} \in [K]^n$ and $\bar{k_0} < u$ we get for all $\bar{k} \in [K]^{n+1}$

$$P(\bar{k}) = j$$

Thus K is also homogenous for P.

∎

Proof of Lemma 32 We start with a proof for the special case $n = 1$. We thus need to find an infinite pre-homogeneous subset for a partition $P : [\omega]^2 \to k$. Consider the complete tree T of fixed branching degree k. In our representation of trees T is the set of all finite sequence of numbers from 0 to $k-1$. For every $s \in T$ we inductively define subsets $A(s) \subseteq \omega$ and numbers $h(s) \in \omega$.

1. $A(\langle\rangle) = \omega$
2. $h(s) = \begin{cases} \text{least element of } A(s) & \text{if } A(s) \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$
3. $A(s^\frown j) = A(s) \cap \{m > h(s) \mid P(\{h(s), m\}) = j\}$ for $0 \leq j < k$.

Let T_0 be the subtree of T with non-empty $A(s)$: $T_0 = \{s \in T \mid A(s) \neq \emptyset\}$. T_0 is an infinite tree. We even argue that there are nodes s of arbitrary length such that $A(s)$ is infinite. This is certainly true fir the empty sequence

$\langle\rangle$ (the root node). If $A(s)$ is infinite we notice that $\bigcup_{0\leq j<k}\{m > h(s) \mid P(\{h(s), m\}) = j\} = \{m > h(s) \mid m \in \omega\}$. Thus

$$\begin{array}{l}[\bigcup_{0\leq j<k}\{m > h(s) \mid P(\{h(s), m\}) = j\}] \cap A(s)\\ =\\ \bigcup_{0\leq j<k}[\{m > h(s) \mid P(\{h(s), m\}) = j\} \cap A(s)]\\ =\\ \bigcup_{0\leq j<k} A(s^\frown j)\end{array}$$

is infinite and one of the sets in this finite union has to be infinite. By König's Lemmas there has to be an infinite path s in T_0. It is easily seen that $\{h(s \restriction n) \mid n \in \omega\}$ is a pre-homogeneous set.

Let us now take up the general case and fix $n \geq 1$ and $k \in \omega$. Notice, that this is not an inductive proof. We consider a more complicated tree that we will again denote by T. In describing T we will deviate from the representation as a collection of finite sequences, as set out in Definition 23. We revert to the more traditional method of giving the node set and the descendant relation of T. The node set of T, named N_T, is the set of all functions $s : [m]^n \to k$ for $m \in \omega$ with $m \geq (n-1)$. t will be a descendant of s, in symbols $s <_T t$, if $s : [m]^n \to k$, $t : [m+1]^n \to k$ and t is an extension of s, i.e. $s \subseteq t$.

Let us explore a bit what T looks like. The root of T is the empty function $s_0 : [n-1]^n \to k$. Note, that $[n-1]^n = \emptyset$. Next observe that $[n]^n$ is the singleton set $\{\{0, \ldots n-1\}\}$. There a k possible descendants P_1^j of s_0 given by $s_1^j(\{0, \ldots n-1\}) = j$. The set $[n+1]^n$ contains, besides $\{0, \ldots n-1\}$, n more subsets of $n+1$ of cardinality n. There are therefore $n * k$ possible descendants for each s_1^j. We see that the branching degree of T gets bigger very fast as we climb the tree, but it will always be finite.

By structural induction we define for all nodes s in N_T the functions $A(s) \subseteq \omega$ and $h(s) \in \omega$:

- $A(s_0) = \omega$
- $h(s) = \begin{cases} \text{least element of } A(s) & \text{if } A(s) \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$
- For $s : [m]^n \to k$, $t : [m+1]^n \to k$ with $s \subseteq t$

$$\begin{aligned} A(t) &= A(s) \cap \{r > h(s) \mid \forall F \in [m+1]^n (P(h(F) \cup \{r\}) = t(F))\} \\ &\quad \text{with } h(F) = \{h(s \restriction [u_1]^n), \ldots, h(s \restriction [u_n]^n)\} \\ &\quad \text{if } F = \{u_1, \ldots, u_n\} \end{aligned}$$

We observe

$$\begin{array}{l} \bigcup_{t \in E(s)} \{r > h(s) \mid \forall F \in [m+1]^n (P(h(F) \cup \{r\}) = t(F))\} \\ = \\ \{r > h(s) \mid r \in \omega\} \\ \text{where } E(s) = \{t : [m+1]^n \to k \mid s \subseteq t\} \end{array}$$

To see this look at $x > h(s)$. Define $t : [m+1]^n \to k$ by $t(F) = P(h(F) \cup \{x\})$ for all $F \in [m+1]^n$. Oviously, $x \in \{r > h(s) \mid \forall F \in [m+1]^n (P(h(F) \cup \{r\}) = t(F))\}$. If $A(s)$ is infinite there the is a descendant t of s such that $A(t)$ is infinite. This shows that the subtree $T_0 = \{s \in T \mid A(s) \neq \emptyset\}$ of T is infinite. By König's Lemma there is an infinite path s in T_0. The set $\{h(s \restriction [m]^n) \mid m \in \omega\}$ is a pre-homogeneous set for P.

∎

2.8.2 Finite Ramsey Theory

Definition 29 *A subset H of natural numbers is called* relatively large *if*

$$card(H) \geq min(H)$$

Thus every infinite subset is relatively large. The set $\{100, 101\}$ is not relatively large, while $\{100, 101, 1\}$ is. The set $\{n \mid 100 \leq n < 200\}$ is another example of a relatively large set.

Lemma 33 *For every natural numbers e, r, k there is a natural number m such that for any partition $P : [m]^e \to r$ there is a relatively large homogenous subset $HG \subseteq m$ of cardinality at least k.*

It has become customary to formulate the statement of this Lemma as

> For every natural numbers e, r, k there is a natural number m such that $m \underset{*}{\to} (k)^e_r$.

Proof We define the set of possible counterexamples

$$
\begin{aligned}
{}^{k}T_{r}^{e} \quad = \quad \{p \mid \quad & \text{there is } m' \in \mathbb{N} \text{ such that } p : [m']^{e} \to r \text{ and} \\
& \text{there is no relatively large homogeneous subset} \\
& H \text{ for } p \text{ with } card(HZ) \geq k\}
\end{aligned}
$$

We define the immediate successor relation $\leq_T$ on ${}^{k}T_{r}^{e}$ by

$$
\begin{aligned}
p_1 \leq_T p_2 \quad \text{iff} \quad & p_1 : [m_1]^{e} \to r, p_2 : [m_1 + 1]^{e} \to r \text{ and} \\
& p_1 = p_2 \restriction [m_1]^{e}
\end{aligned}
$$

Obviously, $({}^{k}T_{r}^{e}, \leq_T)$ is a finitely branching tree.

Assume, for the sake of a contradiction, that the lemma is not true. Thus, for every $m \in \mathbb{N}$ there is some $p : [m]^{e} \to r$ in ${}^{k}T_{r}^{e}$. Therefore, $({}^{k}T_{r}^{e}, \leq_T)$ is an infinite, finitely branching tree. By König's Lemma, Lemma 27, there exists an infinite branch $\{p_i \mid i \in \omega\}$ in this tree. For $p = \bigcup_{i \in \omega} p_i$ we see $p : [\omega]^{e} \to r$. By the infinite Ramsey Theorem, Theorem 31, there is an infinite homogeneous subset $H \subseteq \omega$. Since H is infinite also $\{h \in H \mid h \geq k\}$ is infinite. Let $H_1 \subseteq \{h \in H \mid h \geq k\}$ be a finite subset with $card(H_1) \geq k$ and $m = max(H_1)$. Then H_1 is a relatively large homogeneous set for $p \restriction [m]^{e} \to r$ with $card(H_1) \geq k$. This contradicts the assumption that the lemma is false.

∎

Notice, the unusual pattern of the proof of Lemma 33. Using the assumption that the lemma is false we are able to construct a solution of it. This contradiction establishes the proof.

2.9 Peano Arithmetic with Finite Sets (Optional)

In this section we present and study the typed theory PAFin. This theory uses three types (or sorts) N, S, U where both N and S are subtypes of U, $N \subseteq U$ and $S \subseteq U$. The idea is that N is the type of natural numbers, S is the type of finite sets of natural numbers and U is the least common supertype of both. The signature Σ of PAFin is the union of the signatures Σ_{PA} and Σ_S, that is $\Sigma = \Sigma_{PA} \cup \Sigma_S$.

$\mathbf{\Sigma_{PA}}$:	$\mathbf{\Sigma_S}$:
$0 : N$	$\epsilon : U \times S$
$1 : N$	$\emptyset : S$
$+ : N \times N \to N$	$\{\_, \_\} : U \times U \to S$
$* : N \times N \to N$	$Pot : S \to S$
	$\bigcup : S \to S$
	$card : S \to N$

The theory PAFin will be given by axioms in contrast to a definition by its intended models. The axioms come in two parts, those of Peano Arithmetic and those of Set Theory.

1. $\forall m(m + 1 \neq 0)$
2. $\forall m \forall n(m + 1 \doteq n + 1 \to m \doteq n)$
3. $\forall m(m + 0 \doteq m)$
4. $\forall m \forall n(m + (n + 1) \doteq (m + n) + 1)$
5. $\forall m(m * 0 \doteq 0)$
6. $\forall m \forall n(m * (n + 1) \doteq (m * n) + m)$
7. for every Σ_{PA}-formula $\quad (\phi(0) \wedge \forall n(\phi(n) \to \phi(n + 1))) \to \forall m(\phi)$
8. for every Σ-formula $\quad (\phi(0) \wedge \forall n(\phi(n) \to \phi(n + 1))) \to \forall m(\phi)$

Figure 2.2: Peano Axioms for PAFin

The axioms for $PAFin$ are given in Figures 2.2 and 2.3. Instead of explicit variable declarations of the form e.g., $x : N$ or $x : S$ or $x : U$. We will simplify notation by stipulating

m, n, k	are always of type N
x, y, z	are always of type S
X, Y, Z	are always of type U

A couple of comments are in order. Axioms 1 to 7 are exact copies of the Peano axioms found in many textbooks. The axiom schema 8 makes induction available also for formulas that talk about sets. This is clearly an extension of the usual Peano axiom system since it allows to reason with formulas the are not even present in the syntax of it. By PA we denote the axioms 1 through 7, while PA^S denotes the theory given by all 8 axioms, respectively axioms schemes.

Let $\mathcal{N}$ be the Σ_{PA}- structure $(\mathbb{N}, 0^{\mathcal{N}}, 1^{\mathcal{N}}, +^{\mathcal{N}}, *^{\mathcal{N}})$ with universe the set of natural numbers $\mathbb{N}$ and the usual interpretations of the constants and function symbols. Obviously, $\mathcal{N} \models PA$, i.e., $\mathcal{N} \models \phi$ for all axioms ϕ of PA. $\mathcal{N}$ is called the standard model of PA. But there are also models $\mathcal{M}$ with $\mathcal{M} \models PA$ that are different from $\mathcal{N}$. The theory $PA \cup \{\underbrace{1 + \ldots + 1}_{n \text{ times}} \leq c \mid n \in \mathbb{N}\}$ with the new constant symbol c is consistent, since all its finite subsets are consistent. Thus there is a model $\mathcal{M} \models T$. Obviously, $\mathcal{M}$ is different from $\mathcal{N}$. Models of PA different from $\mathcal{N}$ are callled non-standard models.

Next we will compare the axioms in Figure 2.3 with the axioms of ZF set theory reproduced in Figure 2.1 on page 36. In any case it is essential to pay attention to the type information. In the extensionality axiom a necessary condition of equality is given only for sets since we agreed that variables x,y are always typed $S : x$ and $S : y$. In the foundation axioms the lefthand side of the implication requires that x contains a set y as an element. For sets that exclusively contain natural numbers as elements this axioms does not say anything. Applying the axioms with $\{z\}$ substituted for x we get the usual consequence $\forall z(z \notin z)$. There is no surprise in the subset axiom. We only have to use the variable X of type U, since elements of a set may be natural numbers or again sets. For the following four axioms the PAFin formulation differs on the surface syntax considerably from the ZF axioms. But, on the semantics level they are equivalent. The leading existential quantifiers in all four axioms are eliminated by introducing the corresponding Skolem constant and functions. It is a noteworthy consequence of the given definition of the

A1 Extensionality $\forall X(X \in x \leftrightarrow X \in y) \rightarrow x \doteq y.$

A2 Foundation $\exists y(y \in x) \rightarrow \exists y(y \in x \wedge \forall z \neg(z \in x \wedge z \in y)).$

A3 Subset

$$\exists y \forall X(X \in y \leftrightarrow X \in x \wedge \phi(X)).$$

for any Σ-formula ϕ not containing y.

A4 Empty set $\forall X(X \notin \emptyset).$

A5 Pair set $\forall X \forall Y \forall Z(Z \in \{X, Y\} \leftrightarrow Z \doteq X \vee Z \doteq Y).$

A6 Power set $\forall y \forall Z(Z \in Pot(y) \leftrightarrow \forall U(U \in Z \rightarrow U \in y)).$

A7 Sum $\forall y \forall Z(Z \in \bigcup y \leftrightarrow \exists x(Z \in x \wedge x \in y))$

A8 Finiteness

A8a $card(\emptyset) \doteq 0$

A8b $card(\{X, X\}) \doteq 1$

A8c $\neg \exists U(U \in x \wedge U \in y) \rightarrow$
$card(\bigcup(\{x, y\}) \doteq card(x) + card(y)$

Figure 2.3: Axioms of finite set theory for PAFin

sum operator $\bigcup$ that the sum of a set that contains only natural numbers is $\emptyset$, e.g., $\bigcup\{0, 1, 2, 3, 4\} \doteq \emptyset$. The infinite axioms is dropped. We want the theory of the whole tower of finite set of natual numbers and finiter sets of natural numbers etc. It is not enough to drop the infinity axioms we have to insert axioms that entail that all sets are finite. This is achieved by adding the cardinality function to the signature Σ_S. Since in predicate logic functions are always total this guarantees that all sets have finite cardinality. This argument of courses hinges on the assumption that the three given axioms are enough to force in any interpretation the function symbol *card* to be interpreted unambiguously as the cardinality function. We will come back to this issue later. For the moment we will investigate simple consequences of Peano Arithmetik.

Lemma 34 *The following formulas are derivable from PA.*

1. $\forall n \forall m \forall k((n + m) + k \doteq n + (m + k))$

2. $\forall n \forall m (n + m \doteq m + n)$

3. $\forall x \forall y \forall z (m * (n + k) \doteq m * n + m * k)$

4. $\forall n \forall m \forall k ((n * m) * k \doteq n * (m * k))$

5. $\forall n \forall m (n * m \doteq m * n)$

6. $\forall n \exists m (2 * m \doteq n * (n + 1))$

7. $(1 + 2 + \ldots n) \doteq \frac{1}{2} * n * (n + 1)$

Proof These are easy and well known exercises in inductive proofs.
Let us prove item (6) by induction on n. For $n \doteq 0$ we can choose $m \doteq 0$. Now assume that we know already $n * (n + 1) \doteq 2 * m$.

$$\begin{array}{rcl} (n+1) * (n+2) & \doteq & n * (n+2) + n + 2 \\ & \doteq & n * (n+1) + n + n + 2 \\ & \doteq & 2 * m + 2 * n + 2 \\ & \doteq & 2 * (m + n + 1) \end{array}$$

Item (7) is a favorite exercise for inductive proofs. To stay within the available vocabulary we should have written $2 * (1 + 2 + \ldots n) \doteq n * (n + 1)$. We prefer the more familiar use of the fraction $\frac{1}{2}$ which in the following arguments could be easily avoided. The initial case $n \doteq 0$ in the proof of (7) is obvious. Assume for the induction step that $\Sigma_{i \leq n} i \doteq \frac{1}{2} * n * (n + 1)$.

$$\begin{array}{rcl} \Sigma_{i \leq n+1} i & \doteq & \Sigma_{i \leq n} i + (n+1) \\ & \doteq & \frac{1}{2} * n * (n+1) + (n+1) \\ & \doteq & \frac{1}{2} * (n * (n+1) + 2 * (n+1)) \\ & \doteq & \frac{1}{2} * ((n+2) * (n+1)) \\ & \doteq & \frac{1}{2} * ((n+1) * (n+2)) \end{array}$$

∎

Here are some more easy, but less frequently explicitely mentioned consequences

Lemma 35 *The following formulas are derivable from PA:*

1. $\forall n (n \neq 0 \rightarrow \exists m (n \doteq m + 1))$

2. $\forall n \forall k \forall k'(n + k \doteq n + k' \rightarrow k \doteq k')$

3. $\forall n \forall k \forall k'(n * k \doteq n * k' \wedge n \neq 0 \rightarrow k \doteq k')$

Proof
(1) We use the induction axiom for the formula $\phi(n) = n \neq 0 \rightarrow \exists m(n \doteq m + 1)$. Obviously $PA \vdash \phi(0)$ since the lefthand side of the implication is never true. So assume $n \neq 0 \rightarrow \exists m(n \doteq m + 1)$ and set out to prove $(n + 1) \neq 0 \rightarrow \exists m(n + 1 \doteq m + 1)$. By PA-axiom 1 this is equivalent to $\exists m(n + 1 \doteq m + 1)$. We distinguish two cases.
case 1: $n \doteq 0$ The claim to be proved reads in this case $\exists m(1 \doteq m + 1)$. Obviously, $m \doteq 0$ solves this.
case 1: $n \neq 0$ By inductive assumption we have $\exists m(n \doteq m + 1)$, e.g. $n = m_0 + 1$. By the logical equalitiy axioms we obtain $n + 1 = (m_0 + 1) + 1$, as desired.
(2) This is proved by induction on n.
For $n \doteq 0$ we have to prove $\forall k \forall k'(0 + k \doteq 0 + k' \rightarrow k \doteq k')$ which is obvious by PA axiom 3 and commutativity of $+$.
In the inductive step we assume $\forall k \forall k'(n + k \doteq n + k' \rightarrow k \doteq k')$ and need to prove $\forall k \forall k'((n + 1) + k \doteq (n + 1) + k' \rightarrow k \doteq k')$. By the associative law the claim is equivalent to $\forall k \forall k'((n + (k + 1) \doteq n + (k' + 1) \rightarrow k \doteq k')$. From the induction hypothesis we obtain $k + 1 \doteq k' + 1$. Now, PA axiom 2 yields $k \doteq k'$, as desired.
(3) left to the reader.

∎

The main result of this section will be that the theory $PAFin$ is not stronger than PA. But, it is a long way to get there. We begin by giving a precise meaning when a theory is no stronger than another.

Definition 30 (Conservative Extension) *Let Σ_1, Σ_2 be two signatures with Σ_2 extending Σ_1, i.e. $\Sigma_1 \subseteq \Sigma_2$.*
Let T_i be a theory given by axioms in Σ_i.
T_2 is called a conservative extension *of T_1 iff for any Σ_1-formula ϕ*

$$T_2 \vdash \phi \Leftrightarrow T_1 \vdash \phi$$

In all the cases where we will use the concept of a conservative extension in the following the axioms of T_2 are a superset of the axioms for T_1. But we

could not resists the mathematical habit to give the most general definition without this restriction.

We start with a useful criterion for conservative extensions. The criterion will consider models $\mathcal{M}$ for a signature Σ_2 and use the notation $\mathcal{M} \restriction \Sigma_1$ for $\Sigma_1 \subseteq \Sigma_2$ to denote the structure obtained from $\mathcal{M}$ by omitting the interpretation of all symbols not in Σ_1 . If ϕ is a Σ_1 formula then, obviously, $\mathcal{M} \models \phi$ iff $\mathcal{M} \restriction \Sigma_1 \models \phi$.

Lemma 36 *Let T_i be a theory in signature Σ_i, with $\Sigma_1 \subseteq \Sigma_2$.*
Assume that the following two conditions are satisfied:

1. *For all models $\mathcal{M}$ of T_2 the restriction $\mathcal{M} \restriction \Sigma_1$ is a model of T_1.*
2. *For any model $\mathcal{M}_1$ of T_1 there is a model of $\mathcal{M}_2$ of T_2 such that $\mathcal{M}_2 \restriction \Sigma_1 = \mathcal{M}_1.f$*

Then T_2 is a conservative extension of T_1.

Proof Let ϕ be a Σ_1.
We will first show $T_1 \vdash \phi \Rightarrow T_2 \vdash \phi$. So, assume $T_1 \vdash \phi$. To show $T_2 \vdash \phi$ we consider an arbitrary model $\mathcal{M}$ of T_2. Since by assumption (1) of the lemma we know that $\mathcal{M} \restriction \Sigma_1 \models T_1$ we get from $T_1 \vdash \phi$ the consequence $\mathcal{M} \restriction \Sigma_1 \models \phi$. Thus also $\mathcal{M} \models \phi$.
Now assume conversely that $T_2 \vdash \phi$ with the aim to show $T_1 \vdash \phi$. For any model $\mathcal{M}_1$ of T_1 we thus need to show $\mathcal{M}_1 \models \phi$ By assumption (2) of the lemma there is a Σ_2 structure $\mathcal{M}_2$ with $\mathcal{M}_2 \models T_2$ and $\mathcal{M}_2 \restriction \Sigma_1 = \mathcal{M}_1$. From the case assumption $T_2 \vdash \phi$ we thus get $\mathcal{M}_2 \models \phi$. Since ϕ is a Σ_1 formula this entails $\mathcal{M} \restriction \Sigma_1 \models \phi$. From the second property of $\mathcal{M}_2$ we get $\mathcal{M}_1 \models \phi$.

∎

A class of particularly obvious conservative exentions are definitional extentions.

Definition 31 *Let T_1 be a theory in signature Σ_1. A theory T_2 in signature Σ_2 is called a definitional extention of T_1 if one of the following is true:*

1. *There is a new n-place function symbol f such that*

(a) $\Sigma_2 = \Sigma_1 \cup \{f\}$,

(b) *there is a* Σ_1*-term* t *with variables* $x_1, \ldots, x_n$ *such that* $T_2 = T_1 \cup \{f(x_1, \ldots, x_n) \doteq t\}$

2. *There is a new* n*-place predicate symbol* p *such that*

(a) $\Sigma_2 = \Sigma_1 \cup \{p\}$,

(b) *there is a* Σ_1*-formula* ϕ *with free variables* $x_1, \ldots, x_n$ *such that* $T_2 = T_1 \cup \{p(x_1, \ldots, x_n) \leftrightarrow \phi\}$

3. *There is a new* n*-place function symbol* f *such that*

(a) $\Sigma_2 = \Sigma_1 \cup \{f\}$,

(b) *there is a* Σ_1*-formula* ψ *with variables* $x_1, \ldots, x_n, x_{n+1}$ *such that* $T_2 = T_1 \cup \{f(x_1, \ldots, x_n) \doteq x_{n+1} \leftrightarrow \psi\}$ *and*

(c) $T_1 \vdash \forall x_1, \ldots \forall x_n \exists x_{n+1} \psi$ *and*

Lemma 37 *If* T_2 *is a definitional extension of* T_1 *then* T_2 *is a conservative extension of* T_1.

Proof We will use criterion 36. Since in all three cases $T_1 \subset T_2$ requirement (1) of Lemma 36 is satisfied. Also requirement (2) of Lemma 36 can easily been seen to be true since for any Σ_1 model $\mathcal{M}$ it is obvious how to define the interpretation $f^{\mathcal{M}_2}$ respectively $p^{\mathcal{M}_2}$ in a way such that $\mathcal{M}_2 \models T_2$.

∎

Example 4 *Let* $PAFin_2$ *be the theory obtained from* $PAFin$ *by adding the binary function symbol* $\langle \_, \_ \rangle : U \times U \to S$ *and the defining axiom*

$$\langle X, Y \rangle \doteq \{\{X, X\}, \{X, Y\}\}$$

then $PAFin_2$ *is a definitional extension by part (1) of Definition 31, and hence a conservative, extension of* $PAFin$.
$\langle X, Y \rangle$ *denotes the ordered pair of* X, *and* Y, *see Lemma 3.*

Example 5 *Let PA_2 be the theory obtained from PA by adding the binary predicate symbol $\leq$ and the defining axiom*

$$x \leq y \leftrightarrow \exists z(y \doteq x + z)$$

then PA_2 is a definitional extension by part (2a) of Definition 31, , and hence a conservative, extension of PA.

Example 6 *Let T_1 be an arbitrary theory in signature Σ_1. Further let $\exists x\phi$ be a Σ_1-formula without free variables and c a new constant symbol.*
Let T_2 be the theory in signature $\Sigma_2 = \Sigma_1 \cup \{c\}$ obtain from T_1 by adding the axiom $\exists x\phi(x) \rightarrow \phi(c)$.
Then T_2 is a definitional extension by part (3) of Definition 31, and hence conservative extension of T_1
This is an example of the well-known Skolem extensions. It shows that the item to be defined, here the new constant symbol c, need not be uniquely defined. The existence requirement in this case is $\exists z(\exists x\phi(x) \rightarrow \phi(z))$ which clearly is a tautology.

Example 7 *Let T_1 in signature $\Sigma_1 = \{\leq\}$ be the theory of an order relation and $\Sigma_2 = \Sigma_1 \cup \{a\}$, $T_2 = T_1 \cup \{\forall x(x \leq a)\}$. Then T_2 is not a conservative extension of T_1 since $T_2 \vdash \exists y\forall x(x \leq y)$ but $T_1 \nvdash \exists y\forall x(x \leq y)$.*

Before we go on we note the following easy fact:

Lemma 38 *If T_2 is a conservative extension of T_1 and T_3 is a conservative extension of T_2 then T_3 is a conservative extension of T_1.*

Proof Obvious.

∎

Lemma 39 *If the Σ_2-theory T_2 is a conservative extension of the Σ_2-theory T_1 then there is for every Σ_2-formula ϕ_2 a Σ_1-formula ϕ_1 such that*

$$T_2 \vdash \phi_2 \leftrightarrow \phi_1$$

Proof For definitional extensions of type (1) (2a) of Definition 2a the new function and predicate symbols are simply replaced by their Σ_1 definitions. For definitional extensions of type (3) we use the tautology

$$\forall x(x \doteq t \to \phi(x/t)) \leftrightarrow \phi$$

where $\phi(x/t$ is obtained from ϕ by replacing an occurence of t by the new variable x. If $t = f(t_1, t_2)$ the lefthand side of this formula is logically equivalent to

$$\forall x \forall x_1 \forall x_2((x \doteq f(x_1, x_2) \wedge x_1 \doteq t_1 \wedge x_2 \doteq t_2) \to \phi(x/t))$$

For the new function symbols we can now replace the equation $x \doteq f(x_1, x_2)$ by its Σ_1 definition.
We trust that these hints convince the reader that this works in general.

▪

Lemma 40 *Let $T \supseteq PA$ be a definitional extension in signature Σ of Peano arithmetic PA in $\Sigma \supseteq \Sigma_{PA}$. Then for any Σ-formula*

$$T \vdash (\phi(0) \wedge \forall n(\phi(n) \to \phi(n+1))) \to \forall m(\phi)$$

Proof The induction axioms for formulas ϕ in Σ_{PA} are part of PA and thus of T. All the other new axioms in T are definitions of the new symbols. The claim follows from Lemma 39.

▪

We want to dwell a moment on the conservative extension presented in Example 5.

Lemma 41 *We use $n < m$ as abbreviation for $n \leq m \wedge n \neq m$*

1. $\forall n(0 \geq n \to n \doteq 0)$
2. $\forall n \forall m \forall k(n \leq m \wedge m \leq k \to m \leq k)$
3. $\forall n \forall m \forall k(n < m \to (n + k) < (m + k))$
4. $\forall n \forall m(n < m \vee n = m \vee m < n)$
5. $\forall n \forall m(n \leq m \wedge m \leq n \to n \doteq m)$
6. $\forall n \forall m \forall k(n \leq m \to (n * k) \leq (m * k))$

Proofs
(1) $0 \geq n$ yields by definition $0 \doteq n + m$ for some m. If $m \neq 0$ we get from Lemma 35(1) $m \doteq m_0 + 1$ and thus the contradiction $0 \doteq (n + m_0) + 1$ to PA Axiom 1. Thus $m \doteq 0$ and we get $0 \doteq n + 0 \doteq n$ as desired.
(2) Easy, left to the reader.
(3) From $n < m$ we get by definition $n + n_0 \doteq m$ and $n \neq m$. Thus $(n + k)n_0 \doteq (m + k)$ and $(n + k) \neq (m + k)$. For the last inequality we use Lemma 35 (3). **(4)** The proof proceeds by induction on n. For $m \doteq 0$ we need to show $\forall m(0 < n \vee 0 \neq n \vee m < 0)$. By definition of the order relation $\leq$ and $0+n \doteq n$ we get $0 \leq n$, which is equivalent to $(0 \leq n \wedge 0 \neq n) \vee 0 \doteq n$, i.e., $0 < n \vee 0 \doteq n$.
For the induction step we assume $\forall m(n < m \vee n = m \vee m < n)$ and set out to prove $\forall m((n + 1) < m \vee (n + 1) = m \vee m < (n + 1))$. For $m \doteq 0$ this follows from the initial case of this inductive proof by symmetry. For $m \neq$ ß there is m_0 with $m \doteq m_0 + 1$ by Lemma 35 (1). By induction hypothesis we have $n < m_0 \vee n = m_0 \vee m_0 < n$ by additing 1 to every inequation we get using part (3) $(n + 1) < (m_0 + 1) \vee (n + 1) = (m_0 + 1) \vee (m_0 + 1) < (n + 1)$, i.e. $(n + 1) < m \vee (n + 1) = m \vee m < (n + 1)$, as desired.
(5) By definition we get from $n \leq m \wedge m \leq n$ numbers n_0, m_0 with $n+n_0 \doteq m$ and $m + m_0 \doteq n$. Thus $n \doteq n + n_0 + m_0$. From Lemma 35 (2) we get $0 \doteq n_0 + m_0$ Part (1) of Lemma 35 can than be used to derive $0 \doteq n_0$ and $0 \doteq m_0$.
(6) From $m \doteq n + n_0$ we get $(m * k) \doteq (n + n_0) * k \doteq n * k + n_0 * k$, i.e., $n * k \leq (m * k)$.

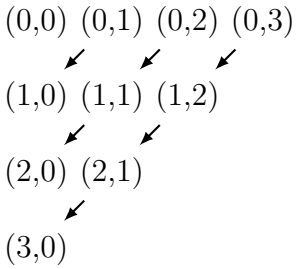
∎

The examples for conservative extentsions we have seen so far were only a warm-up for the following stunning result:

Theorem 42 *$PAFin$ is a conservative extension of PA.*

The proof will keep us busy for the rest of this section. We start slow with simple facts and move on to more and more complex formulas that are derivable in PA.

As a preparation for the proof of the next lemma we recall the notorious enumeration of pairs of natural numbers:



We want to find a formula that computes the position of (m, n) in this enumeration starting with position 0. We observe that the diagonals contain all pairs with the same sum of its components and each diagonal is by one greater than the previous one. The diagonal starting in $(0, k)$ contains $k+1$ pairs. The position of $(0, m)$ thus equals $1+2+\ldots m$ which by Lemma 34 (7) equals $\frac{1}{2}*m*(m+1)$. To compute the position of (m, n) we first compute the position of $(0, m+n)$ and then add m. We obtain $\frac{1}{2}*(m+n)*(m+n+1)+m$.

These considerations show

$\mathcal{N} \models \forall k \exists m \exists n (k \doteq \frac{1}{2} * (m+n) * (m+n+1) + m)$

$\mathcal{N} \models \forall m \forall n \forall m' \forall n' ($
$\frac{1}{2} * (m+n) * (m+n+1) + m \doteq \frac{1}{2} * (m'+n') * (m'+n'+1) + m'$
$\rightarrow m \doteq m' \wedge n \doteq n')$

But, are these two formulas derivable from PA? We have to work a bit harder to see this.

Lemma 43 *The following formulas are derivable in PA.*

1. $\forall k \exists m \exists n (k \doteq \frac{1}{2} * (m+n) * (m+n+1) + m)$
2. $\forall m \forall n \forall m' \forall n' ($
$\frac{1}{2} * (m+n) * (m+n+1) + m \doteq \frac{1}{2} * (m'+n') * (m'+n'+1) + m'$
$\rightarrow m \doteq m' \wedge n \doteq n')$

Proof
(1) We use, of course, induction on k. For $k \doteq 0$ obviously $m \doteq n \doteq 0$ does the job. So assume $k \doteq \frac{1}{2} * (m+n) * (m+n+1) + m$. We want to find m', n' such that $k+1 \doteq \frac{1}{2} * (m'+n') * (m'+n'+1) + m'$.

Case $n > 0$: Set $m' \doteq m + 1$ and $n' \doteq n - 1$ and compute

$$\begin{array}{rcl} \frac{1}{2} * (m' + n') * (m' + n' + 1) + m' & \doteq & \frac{1}{2} * (m + n) * (m + n + 1) + m + 1 \\ & \doteq & k + 1 \end{array}$$

Case $n \doteq 0$: Thus $k \doteq \frac{1}{2} * m * (m + 1) + m$.
Set $m' \doteq 0$ and $n' \doteq m + 1$ and compute

$$\begin{array}{rcl} \frac{1}{2} * (m' + n') * (m' + n' + 1) + m' & \doteq & \frac{1}{2} * (m + 1) * (m + 2) \\ & \doteq & \frac{1}{2} * (m * (m + 1) + 2 * (m + 1)) \\ & \doteq & \frac{1}{2} * m * (m + 1) + m + 1) \\ & \doteq & k + 1 \end{array}$$

(2) The trick is to prove

$$\begin{array}{l} \forall k \forall m \forall n \forall m' \forall n' (\\ k \doteq \frac{1}{2} * (m + n) * (m + n + 1) + m \doteq \frac{1}{2} * (m' + n') * (m' + n' + 1) + m' \\ \rightarrow m \doteq m' \wedge n \doteq n') \end{array}$$

by induction on k. The case $k \doteq 0$ is easy. For the inductive step a replacement of (m, n) by (m', n') as in part (1) of the proof will succeed. The details are left to the reader.

∎

Lemma 44 *Let Σ^1_{PA} bew the extension of Σ_{PA} by the following function symbols:*
$[\_, \_] : N \times N \rightarrow N$
$\pi_1 : N \rightarrow N$
$\pi_2 : N \rightarrow N$
Let PA^1 the the extension of PA by the axioms

1 $\forall k \forall m(\pi_1(k) \doteq m \leftrightarrow \exists n(2 * k \doteq (m + n) * (m + n + 1) + 2 * m)$
2 $\forall k \forall n(\pi_2(k) \doteq n \leftrightarrow \exists m(2 * k \doteq (n + m) * (n + m + 1) + 2 * m)$
3 $\forall m \forall n \forall k([n, m] \doteq k \leftrightarrow (2 * k \doteq (m + n) * (m + n + 1) + 2 * m))$

We think of $[n, m]$ as a code for the order pair of n and m and of π_1, π_2 as the first respectively second projection.
Furthermore

$$PA^1 \vdash \forall m \forall n(\pi_1([m, n]\ ) \doteq n \wedge \pi_2([m, n]\ ) \doteq n)$$

Proof We claim that the addition of the function symbols π_1, π_2 and $[\_,\_]$ yields definitional extensions of type (3) from Definition 31. We only need to show that part (c) of this definition is satisfied. The formula to be proved for π_1, π_2 is in both cases

$$\forall k \exists m \exists n (k \doteq \frac{1}{2} * (m+n) * (m+n+1) + m)$$

which is Lemma 43 (1).
For $[\_,\_]$ we need to show that $\forall m \forall n \exists k ((2*k \doteq (m+n)*(m+n+1)+2*m))$ which is clear by Lemma 34 (6).

Derivability of the formula $\forall m \forall n (\pi_1([m,n]\,) \doteq n \wedge \pi_2([m,n]\,) \doteq n)$ follows from Lemma 43 (2).

∎

We have seen the power of the induction axioms of PA. Sometimes however, another proof principle, called the least number principle, is more convenient.

Lemma 45 *For any Σ-formula ϕ the following is derivable in PA:*

$$\exists n \phi \rightarrow \exists n (\phi \wedge \forall m (m < n \rightarrow \neg \phi))$$

Proof Given $\phi(n)$ define $\phi'(n) = \exists k (k \leq n \wedge \phi(k))$.
It can be easily seen that $\forall n (\phi'(n) \rightarrow \forall m (n \leq m \rightarrow \phi'(m)))$ is derivable from PA. The induction axiom for $\neg\phi'(n)$ reads

$$(\neg\phi'(0) \wedge \forall n (\neg\phi'(n) \rightarrow \neg\phi'(n+1)) \rightarrow \forall n (\neg\phi'(n))$$

Contraposition yields

$$\exists n (\phi'(n)) \rightarrow (\phi'(0) \vee \exists n (\neg\phi'(n) \wedge \phi'(n+1))$$

By definition of ϕ' the following formulas are derivable from PA.

1 $\phi'(0) \rightarrow \phi(0)$
2 $\forall n ((\neg\phi'(n) \wedge \phi'(n+1)) \rightarrow \phi(n+1) \wedge \forall m (m < n \rightarrow \neg\phi(m)))$

From these the claim of the lemma follows directly.

∎

For the next developments we need the notion of divisibility.

Definition 32 *We say that n is a divisor of m (or n divides m) if there is k with $n * k \doteq m$.*
This gives rise to a definitional extension that adds the new symbol $n \mid m$ defined by $\exists k(n * k \doteq m)$.

Lemma 46 *Adding the symbol $\_ - \_$ and the axioms for truncated difference*

$$\begin{array}{l} \forall n \forall m(m > n \to (m - n) + n \doteq m) \\ \forall n \forall m(m \leq n \to m - n \doteq 0) \end{array} \tag{2.3}$$

is a definitional extenstion of PA.
Teh following are derivable

1. $\forall n \forall m \forall k(k * (m - n) \doteq k * m - k * n)$

Proof
The formula that needs to be derivable according to Definition 31 (2a) is an ewasy consequence of the definition of $m \geq n$.
(1) The cases $m \leq n$ and $k \doteq 0$ are simple. If $m \geq n$ and $k \neq 0$ then $k * m \geq k * n$. We obtain by distributivity and the truncated difference axiom for m and n:
$k * (m - n) + k * n \doteq k * ((m - n) + n) \doteq k * m$
From the truncated difference axioms for $k * m$ and $k * n$ we also get: $(k * m - k * n) + k * n \doteq k * m$
Thus Lemma 35 (2) gives $(k * m - k * n) \doteq k * (m - n) + k * n$.

∎

Lemma 47 *The following formulas are derivable in the definitional extension of PA.*

1. $\forall n \forall m_1 \forall m_2(n \mid m_1 \wedge n \mid m_2 \to n \mid m_1 + m_2)$
2. $\forall n \forall m_1 \forall m_2(n \mid m_1 \wedge n \mid m_2 \wedge m_2 \geq m_1 \to n \mid (m_2 - m_1))$

Proof
(1) This is fairly obvious: From $n * k_1 \doteq m_1$ and $n * k_2 \doteq m_2$ we obtain $n * (k_1 + k_2) \doteq m_1 + m_2$ by the distributive law.
(2) By definition of divisibility we get numbers n_1 and n_2 such that $m_1 \doteq n * n_1$ and $m_2 \doteq n * n_2$. From Lemma 41 (4) we know $n_1 < n_2 \vee n_1 \doteq n_2 \vee n_2 < n_1$. From $n_2 < n_1$ we obtain from Lemma 41 (6) $m_1 \doteq n * n_2 < n * n_1 \doteq m_2$ which contradicts $m_2 \geq m_1$ by Lemma 41 (5). Thus we must have $n_1 < n_2 \vee n_1 \doteq n_2$. If $n_1 \doteq n_2$ then also $m_1 \doteq m_2$. From $m_2 \doteq m_1 + 0$ and Lemma 35 (2) we obtain $(m_2 - m_1) \doteq 0$ and thus $n \mid (m_2 - m_1)$ trivially.
The last case to be considered is $n_1 < n_2$. By definition there is n' with $n_1 + n' \doteq n_2$.

$m_2 \doteq m_1 + (m_2 - m_1)$	assumption in formula (2)
$n * n_2 \doteq n * n_1 + (m_2 - m_1)$	assumptions in formula (2)
$n * (n_1 + n') \doteq n * n_1 + (m_2 - m_1)$	case assumption $n_1 < n_2$
$n * n_1 + n * n' \doteq n * n_1 + (m_2 - m_1)$	distributive law
$n * n' \doteq (m_2 - m_1)$	injectivity of $+$, Lemma 35 (2)
$n \mid (m_2 - m_1)$	definition of divisibility

∎

Lemma 48 *Assume that the only common divisor of n and m is 1. Then there exists n_1, n_2, m_1, m_2 such that $n * n_1 = m * m_1 + 1$ and $m * m_2 = n * n_2 + 1$.*

Proof
The claim follows from the well-known algorithm of Euclid to compute the greatest common devisor of two integers. To keep these notes self-contained we include the proof, as far as we need it, here. The algorithm is usually presented for integers. We adapt it to natural numbers, i.e., positive integers.

We define pairs of natural numbers a_i,b_i for $0 \leq i < r$ by the following rules

1. $a_0 \doteq max(n, m)$, $b_0 \doteq min(n, m)$
2. $a_i > b_i$ for all i with $0 \leq i < n$
3. $\{a_{i+1}, b_{i+1}\} \doteq \{b_i, a_i - b_i\}$
4. $a_r \doteq b_r$

First we note that from $a_i > b_i$ and $a_i > a_i - b_i$ we get $max(a_i, b_i) > max(a_{i+1}, b_{i+1})$. For some j we must there get $max(a_j, b_j) \doteq 0$. This entails $a_{j-1} - b_{j-1} \doteq 0$, i.e., $a_{j-1} \doteq b_{j-1}$. We may thus stop the construction with $r \doteq j - 1$.
For $n \doteq 14$, $m \doteq 9$ we obtain

a_0	$\doteq$	14	b_0	$\doteq$	9	a_0	$\doteq$	n	b_0	$\doteq$	m
a_1	$\doteq$	9	b_1	$\doteq$	5	a_1	$\doteq$	b_0	b_1	$\doteq$	$a_0 - b_0$
a_2	$\doteq$	5	b_2	$\doteq$	4	a_2	$\doteq$	b_1	b_2	$\doteq$	$a_1 - b_1$
a_3	$\doteq$	4	b_3	$\doteq$	1	a_3	$\doteq$	b_2	b_3	$\doteq$	$a_2 - b_2$
a_4	$\doteq$	3	b_4	$\doteq$	1	a_4	$\doteq$	$a_3 - b_3$	b_4	$\doteq$	b_3
a_5	$\doteq$	2	b_5	$\doteq$	1	a_5	$\doteq$	$a_4 - b_4$	b_5	$\doteq$	b_4
a_6	$\doteq$	1	b_6	$\doteq$	1	a_6	$\doteq$	$a_5 - b_5$	b_6	$\doteq$	b_5

If k is a common divisor of a and b we obtain by repeated application of Lemma 47 (2) starting with $i \doteq 0$ that k is a common divisor of a_i and b_i for all $0 \leq i \leq r$. Using Lemma 47 (1) starting with $i \doteq r$ we see that a_r is a common divisor of a_i and b_i for all $0 \leq i \leq r$. Since by assumption of the lemma the only common divisor of n and m is 1 we must have $a_r \doteq b_r \doteq 1$.

Substituting the defining equations for the sequence a_i, b_i from bottom to top starting with $a_r \doteq 1$ and $b_r \doteq 1$ we obtain for every i, $r > i \geq 0$

$$\begin{array}{l} n_1^i * a_i \doteq m_1^i * b_i + 1 \quad \text{and} \quad m_2^i * b_i \doteq n_2^i * a_i + 1 \\ \text{or} \\ m_1^i * b_i \doteq n_1^i * a_i + 1 \quad \text{and} \quad n_2^i * a_i \doteq m_2^i * b_i + 1 \end{array} \tag{2.4}$$

Before we prove this lets us look at the above example computation:

		defining equations					version 1				version 2		
a_0	$\doteq$	n	b_0	$\doteq$	m		$11 * m - 7 * n$	$\doteq$	1		$2 * n - 3 * m$	$\doteq$	1
a_1	$\doteq$	b_0	b_1	$\doteq$	$a_0 - b_0$		$4 * a_1 - 7 * b_1$	$\doteq$	1		$2 * b_1 - a_1$	$\doteq$	1
a_2	$\doteq$	b_1	b_2	$\doteq$	$a_1 - b_1$		$4 * b_2 - 3 * a_2$	$\doteq$	1		$a_2 - b_2$	$\doteq$	1
a_3	$\doteq$	b_2	b_3	$\doteq$	$a_2 - b_2$		$a_3 - 3 * b_3$	$\doteq$	1		b_3	$\doteq$	1
a_4	$\doteq$	$a_3 - b_3$	b_4	$\doteq$	b_3		$a_4 - 2 * b_4$	$\doteq$	1		b_4	$\doteq$	1
a_5	$\doteq$	$a_4 - b_4$	b_5	$\doteq$	b_4		$a_5 - b_5$	$\doteq$	1		b_5	$\doteq$	1
a_6	$\doteq$	$a_5 - b_5$	1	$\doteq$	b_5		a_6	$\doteq$	1		b_6	$\doteq$	1

Claim 2.4 is proved by reverse induction from $i \doteq r$ to $i \doteq 0$. (If you dont't like this to forward induction on $j \doteq r - i$ from $j \doteq 0$ to $j \doteq r$.)
For $i \doteq r$ we set $n_1^r \doteq 1$, $m_1^r \doteq 0$, $n_2^r \doteq 0$ and $m_2^r \doteq 1$.

Now assume

$$n_1^{i+1} * a_{i+1} \doteq m_1^{i+1} * b_{i+1} + 1 \quad \text{and} \quad m_2^{i+1} * b_{i+1} \doteq n_2^{i+1} * a_{i+1} + 1$$

or

$$m_1^{i+1} * b_{i+1} \doteq n_1^{i+1} * a_{i+1} + 1 \quad \text{and} \quad n_2^{i+1} * a_{i+1} \doteq m_2^{i+1} * b_{i+1} + 1$$

Case 1 $b_{i+1} \doteq b_i$ and $a_{i+1} \doteq a_i - b_i$ leads to the following computation

$$\begin{array}{ll}
n_1^{i+1} * a_{i+1} \doteq m_1^{i+1} * b_{i+1} + 1 & \text{and} \quad m_2^{i+1} * b_{i+1} \doteq n_2^{i+1} * a_{i+1} + 1 \\
n_1^{i+1} * (a_i - b_i) \doteq m_1^{i+1} * b_i + 1 & \text{and} \quad m_2^{i+1} * b_i \doteq n_2^{i+1} * (a_i - b_i) + 1 \\
n_1^{i+1} * a_i - n_1^{i+1} * b_i \doteq m_1^{i+1} * b_i + 1 & \text{and} \quad m_2^{i+1} * b_i \doteq (n_2^{i+1} * a_i - n_2^{i+1} * b_i) + 1 \\
n_1^{i+1} * a_i \doteq (m_1^{i+1} + n_1^{i+1}) * b_i + 1 & \text{and} \quad (m_2^{i+1} + n_2^{i+1}) * b_i \doteq n_2^{i+1} * a_i + 1 \\
n_1^i * a_i \doteq m_1^i * b_i + 1 & \text{and} \quad m_2^i * b_i \doteq n_2^i * a_i + 1 \\
\text{or} & \\
m_1^{i+1} * b_{i+1} \doteq n_1^{i+1} * a_{i+1} + 1 & \text{and} \quad n_2^{i+1} * a_{i+1} \doteq m_2^{i+1} * b_{i+1} + 1 \\
m_1^{i+1} * b_i \doteq n_1^{i+1} * (a_i - b_i) + 1 & \text{and} \quad n_2^{i+1} * (a_i - b_i) \doteq m_2^{i+1} * b_i + 1 \\
m_1^{i+1} * b_i \doteq (n_1^{i+1} * a_i - n_1^{i+1} * b_i) + 1 & \text{and} \quad n_2^{i+1} * a_i - n_2^{i+1} * b_i \doteq m_2^{i+1} * b_i + 1 \\
(m_1^{i+1} + n_1^{i+1}) * b_i \doteq n_1^{i+1} * a_i + 1 & \text{and} \quad n_2^{i+1} * a_i \doteq (m_2^{i+1} + n_2^{i+1}) * b_i + 1 \\
m_1^i * b_i \doteq n_1^i * a_i + 1 & \text{and} \quad n_2^i * a_i \doteq m_2^i * b_i + 1
\end{array}$$

Case 2 $a_{i+1} \doteq b_i$ and $b_{i+1} \doteq a_i - b_i$ leads to the following computation

$$\begin{array}{ll}
n_1^{i+1} * a_{i+1} \doteq m_1^{i+1} * b_{i+1} + 1 & \text{and} \quad m_2^{i+1} * b_{i+1} \doteq n_2^{i+1} * a_{i+1} + 1 \\
n_1^{i+1} * b_i \doteq m_1^{i+1} * (a_i - b_i) + 1 & \text{and} \quad m_2^{i+1} * (a_i - b_i) \doteq n_2^{i+1} * b_i + 1 \\
n_1^{i+1} * b_i \doteq (m_1^{i+1} * a_i - m_1^{i+1} * b_i) + 1 & \text{and} \quad m_2^{i+1} * a_i - m_2^{i+1} * b_i \doteq n_2^{i+1} * b_i + 1 \\
(n_1^{i+1} + m_1^{i+1}) * b_i \doteq m_1^{i+1} * a_i + 1 & \text{and} \quad m_2^{i+1} * a_i \doteq (n_2^{i+1} + m_2^{i+1}) * b_i + 1 \\
m_1^i * b_i \doteq n_1^i * a_i + 1 & \text{and} \quad n_2^i * a_i \doteq m_2^i * b_i + 1 \\
\text{or} & \\
m_1^{i+1} * b_{i+1} \doteq n_1^{i+1} * a_{i+1} + 1 & \text{and} \quad n_2^{i+1} * a_{i+1} \doteq m_2^{i+1} * b_{i+1} + 1 \\
m_1^{i+1} * (a_i - b_i) \doteq n_1^{i+1} * b_i + 1 & \text{and} \quad n_2^{i+1} * b_i \doteq m_2^{i+1} * (a_i - b_i) + 1 \\
m_1^{i+1} * a_i - m_1^{i+1} * b_i \doteq n_1^{i+1} * b_i + 1 & \text{and} \quad n_2^{i+1} * b_i \doteq (m_2^{i+1} * a_i - m_2^{i+1} * b_i) + 1 \\
m_1^{i+1} * a_i \doteq (n_1^{i+1} + m_1^{i+1}) * b_i + 1 & \text{and} \quad (n_2^{i+1} + m_2^{i+1}) * b_i \doteq m_2^{i+1} * a_i + 1 \\
n_1^i * a_i \doteq m_1^i * b_i + 1 & \text{and} \quad m_2^i * b_i \doteq n_2^i * a_i + 1
\end{array}$$

■

We do not need much of the theory of prime numbers, but we cannot do completely without them.

Definition 33 *A natural number $n > 1$ is called a* prime number *if its only divisors are* 1 *and* n.
Thus 2 *is the smallest prime number.*

Lemma 49 *Let p be a prime number.*
If $p \mid a * b$ then $p \mid a$ or $p \mid b$.

Proof By assumption we know $a * b \doteq k * p$ for some k. Assume that p does not divide a. Then the only common divisor of p and a is 1. By Lemma 48 we get m and n with $n * p \doteq m * a + 1$. Multiplying both sides of this equation with b we get $n * p * b \doteq m * a * b + b$. Substituting the equation for $a * b$ we obtain $n * p * b \doteq m * k * p + b$. Now, Lemma 47 (2) gives us $p \mid b$.

∎

Lemma 50 *The new binary function symbol* $\mathrm{seq}(\_,\_)$ *is added in a definitional extension of PA by the definition:*

$$\begin{aligned} \mathrm{seq}(m,k) \doteq n \quad \leftrightarrow \quad & \exists m'(\pi_1(m) \doteq m' * (1 + \pi_2(m) * (k+1)) + n) \\ & \wedge 0 \leq n \wedge n < (1 + \pi_2(m) * (k+1)) \end{aligned}$$

Using a more familiar notation, that we have not introduced here, we could write $\mathrm{seq}(m,k) \doteq \pi_1(m) mod (1 + \pi_2(m) * (k+1))$.

Proof To show that adding $\mathrm{seq}(\_,\_)$ with tha given definition is a definitional extension according to Lemma 31 (3) it siffices to show that

$$\forall m \forall n \exists k \exists m'(n \neq 0 \rightarrow m \doteq m' * n + k \wedge 0 \leq k \wedge k < n)$$

is derivable from PA. For $m \doteq 0$ the statement is true if we choose $m' \doteq k \doteq 0$ for any n. The inductive proof turns out to be easy. Assume that the claim is true for m and all $n \neq 0$ and we want to show it for $m+1$ and all $n \neq 0$. By induction hypothesis we have m' and k satisfying

$$m \doteq m' * n + k \wedge 0 \leq k \wedge k < n).$$

Then

$$m + 1 \doteq \begin{cases} m' * n + k + 1 & \text{if } k + 1 < n \\ (m'+1) * n + 0 & \text{if } k + 1 \doteq n \end{cases}$$

We leave the easy computations to the reader.

∎

Lemma 51 *For following formulas are derivable for every k*

$$Seq_k = \forall n_0 \ldots \forall n_{k-1} \exists m(\bigwedge_{0 \leq i < k} seq(m,i) \doteq n_i)$$

Proof As a preparatory step we consider the numbers $1 + n!(i + 1)$ for $0 \leq i \leq n$, with $n! = 1 * 2 * \ldots * n$ as usual.

$$\begin{array}{l}\text{For } 0 \leq i < j \leq n \text{ the greatest common divisor of} \\ 1 + n!(i + 1) \text{ and } 1 + n!(j + 1) \text{ is } 1.\end{array} \tag{2.5}$$

Proof of (2.5): If k is a prime divisor of $1+n!(i+1)$ and $1+n!(j+1)$ then by Lemma 47 (2) k is also a divisor of the difference $n!(j-i)$. By Lemma 49 k is a divisor of $n!$ or $(j-i)$. Since $1 \leq (j-i) \leq n$ the difference $(j-i)$ is a factor of $n!$. Thus we see that k is a divisor of $n!$ in any case and therefore a divisor of $n!(i+1)$. Again appealing to Lemma 47 (2) we see that k is a divisor of the difference $(1 + n!(i + 1)) - n!(i + 1) \doteq 1$. This yields $k \doteq 1$.

`to be completed`

2.10 Comments

1. ZF is by far the most common axiom system for set theory. Others are the Neumann-Bernays-Gödel system (this is e.g. used in [Rubin, 1967]) and the Taski-Grothendiek system.

2. Notice, that ZF set theory is a theory of first-order logic, despite the fact that sets are involved, which are usually thought of as second-order objects. The point here is, that the classification into second-order, third-order and so on is relative to a fixed level of first-order elements. In set theory sets are first-order elements.

3. There are versions of set theory that start out with an initial set of elements of arbitrary kind, usually called *urelements*. On top of these set theory is built, i.e. there will be set of urelements, sets of sets of urelements and so on. In our exposition we are interested in reduction to first principles, so it makes sense to go all the way and consider nothing but sets.

4. We have chosen the textbook [Takeuti & Zaring, 1971]as a reference mainly for the reason that it is explicitely mention in the ANSI standard draft for **Z**. A very gentle, but rigorous introduction may be found in [Halmos, 1994, Halmos, 1974].

2.11 Exercises

Exercise 2.11.1 *Is the formula* $\forall x(x \neq \emptyset \rightarrow \exists y(y \in x \wedge x \cap y = \emptyset)$ *derivable from the ZF axioms?*

Exercise 2.11.2

1. *Show by induction on* m
   $\forall m \forall n(n \in m \rightarrow n^+ \in m \vee n^+ = m)$

2. *Show by induction on* n
   $\forall n \forall m(n \in m \vee n = m \vee m \in n)$

Exercise 2.11.3 *Consider a finite set a and prove:*
If $\bigcup a = a$ then $a = \emptyset$.

Exercise 2.11.4

1. *Give an example of a transitive set, that is not an ordinal.*
2. *Prove: if a is an ordinal then a^+ is also.*

Exercise 2.11.5 *Show that the following class terms are not sets.*

1. $\{x \mid x \notin x\}$
2. $\{x \mid x = x\}$
3. $\{x \mid rel(x)\}$

Exercise 2.11.6 *Show on the basis of the axioms of ZF set theory that for any two sets a, b the cartesian product $a \times b = \{\langle z_1, z_2\rangle \mid z_1 \in a \wedge z_2 \in b\}$ is also a set.*

Exercise 2.11.7 *Show that the class Ord of all ordinals is not a set.*

Exercise 2.11.8 *Let x be a subset of the class of all ordinals. By Lemma 24(4) $\alpha = \bigcup x$ is again an ordinal. It can easily be seen that $\alpha \leq \beta \Leftrightarrow_{def} \alpha \subseteq \beta$ defines a linear order on Ord.*
Show that α is the least upper bound of x, i.e.

1 $\beta \leq \alpha$ *for all* $\beta \in x$
2 *if* $\beta \leq \gamma$ *for all* $\beta \in x$, *then* $\alpha \leq \gamma$

Exercise 2.11.9 *Can there be a set a such that $a \times a = \mathcal{P}(a)$?*
For the definition of the cartesion product see Exercise 2.11.6.

Exercise 2.11.10 *This exercise I owe to a participant of my course. Can there be four relations s, p, q, r such that*

$$\langle q, s\rangle \in r \quad \textit{and} \quad \langle p, r\rangle \in s \ ?$$

Background: The problem originates from a foundational context where every existing object is a relation. Here is an attempt to a positive answer. Let the relation r be defined by $r(x, y) \Leftrightarrow x$ is a subrelation of y. Furthermore let s be the universal relation, i.e., $s(x, y)$ for x, y. Then the requirements seem to be satisfied?

Exercise 2.11.11 *Show, that there can be no set a with the property $a = \{a, \emptyset\}$.*
In other words, the formula $\neg\exists x(\forall z(z \in x \leftrightarrow z = x \vee z = \emptyset))$ is derivable from the axioms.

Exercise 2.11.12 *Show that for no natural number n can there be sets $a_1, \ldots a_n$ with*

$$a_1 \in a_2 \in \ldots \in a_{n-1} \in a_n \in a_1$$

Exercise 2.11.13 *Let X be a subclass of the class of all ordinals Ord such that*

1. $0 \in X$
2. *if $\alpha \in X$ then $\alpha^+ \in X$*
3. *if $\forall\beta(\beta \in \alpha \to \beta \in X)$ then $\alpha \in X$.*

Then $X = Ord$.
This fact is called the principle of ordinal induction.

Exercise 2.11.14 *Let G be the set of all finite sequences of elements from $\{0, 1\}$. In our setting G is the set of functions s with $range(s) \subseteq \{0, 1\}$ and $dom(s) \in \mathbb{N}$. We define the usual* lexicigraphic *ordering $<_{lex}$ on G:*

$$s <_{lex} t \Leftrightarrow \begin{cases} s \text{ is an initial sequence of } t \text{ or} \\ \exists i(0 \leq i < \ length(s) \wedge i < \ length(t) \wedge \\ \quad \forall j(0 \leq j < i \to s_j = t_j) \wedge s_i < t_i) \end{cases}$$

Is $(G, <_{lex})$ wellordered? (see Definition 17)

Exercise 2.11.15 *Let α, β be ordinals. Then*

$$(\alpha, \epsilon) \cong (\beta, \epsilon) \Rightarrow \alpha = \beta$$

Exercise 2.11.16 *Show that the class term $\{x \mid ord(x)\}$ is not a set.*

Exercise 2.11.17

1. *Let s be the disjoint union of two finite sets s_1, s_2, i.e., $s = s_1 \cup s_2$ and $\neg\exists x(x \in s_1 \wedge x \in s_2)$.*
   Then s is again finite.

2. *Let s be again the disjoint union of two sets s_1, s_2, we write $s = s_1 \uplus s_2$. If s is infinite then s_1 or s_2 is infinite. This includes, of course the case where both s_1 and s_2 are infinite.*

3. *Generalize part (1) of this exercise to finitely many finie sets. More precisely: given a finite set a*

   (a) *such that every element of a is a finite set,*
   $\forall s(s \in a \rightarrow fin(s))$

   (b) *the elements of a are mutually disjoint,*
   $\forall s_1 \forall s_1(s_1 \in a \wedge s_2 \in a \rightarrow s_1 \cap s_2 = \emptyset)$

   then $\bigcup a$ is finite.
   Less formaly we could claim: if $s_0, \ldots, s_{k-1}$ are mutually disjount finite sets than also $s_0 \cup \ldots \cup s_{k-1}$ is finite. This correlates with the previous formulation via $s = \{s_0, \ldots, s_{k-1}\}$ and $k = card(a)$.

4. *If the union of finitely many mutually disjoint sets $s_1, \ldots, s_k$ is infinite, then at least one s_i is infinite.*

Chapter 3

Modal Logic

Modal logic is not only concerned with the study of truth or falsity of propositions but also takes into account in which way, in which modality, a proposition is considered to be true or false. In philosophical logic the distinction between statements that are *neccessarily true* and those that are *possibly true* was important. Other modalities could be time where one would distinguish between facts that are true today and those that are true tomorrow or were true yesterday. Another variation in modal logic is to reason about statements a person believes to be true, or knows to be true.

Contributions to philosophical modal logic date back to the Middle Ages and even into Antiquity, with the unavoidable reference to Aristotle. There is great agreement in the scientific community to regard the publication [Lewis, 1918] by C. I. Lewis as the beginning of the new era of formal study of modal logic. More recently modal logic was discovered as a useful theory for computer science, in particular in the areas of artificial intelligence, temporal reasoning and program semantics.

Hughes und Cresswell could still claim that their book [Hughes & Cresswell, 1972] published in 1968 covers the collected knowledge on modal logics of their time. Today it is simply impossible to present and overview of the field with any contention to completeness. Here is a list of selected monographs and survey articles not devoted to a specific topic within modal logic and accessible without prior knowledge.

- Melvin Fitting's chapter in the Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 1, [Fitting, 1993],
- Part One in the CSLI Lecture Notes by Robert Goldblatt, [Goldblatt, 1987],
- Colin Stirling's chapter in the second volume of the Handbook of Logic in Computer Science, [Sterling, 1992],
- the chapter on propositional modal logic by Robert Bull und Krister Segerberg and the chapter on modal quantifier logic by James Garson in Volume II of the Handbook of Philosophical Logic, [Bull & Segerberg, 1984], [Garson, 1984],
- the successor of their 68-er book by Hughes and Cresswell, [Hughes & Cresswell, 1984],

- the chapter by Peter Steinacker in *Einführung in die Nichtklassische Logik*, Akademie-Verlags, Berlin, [Steinacker, 1990]

3.1 Syntax and Semantics

The study of modal logic centers on propositional modal logics. First-order modal logics play a much lesser role. In this section we will present the basic definitions of propositional modal logic: syntax, semantics, tautologies and logical inference.

Definition 34 (Syntax of Modal Logic) *Given a set of propositional variables* PVar *we define the set $PModFml$ of propositional modal formulas inductively by*

1. PVar $\subset PModFml$, *i.e. every propositional variable is a model formula,*
2. *if $F_1, F_2 \in PModFml$ then also*
   $F_1 \wedge F_2$, $F_1 \vee F_2$, $F_1 \rightarrow F_2$ *and* $\neg F_1$ *in* $PModFml$,
3. *if $F \in PModFml$ then also $\Box F \in PModFml$ and $\Diamond F \in PModFml$.*

A useful metric for modal formulas is the maximal nesting of modal operators.

Definition 35 (Modal Depth) *The* model depth $md(F)$ *of a formula $F \in PModFml$ is inductively defined:*

1. $md(p) = 0$ *for* $p \in$ PVar,
2. $md(\neg F) = md(F)$,
3. $md(F_1 \wedge F_2) = md(F_1 \vee F_2) = max\{md(F_1), md(F_2)\}$,
4. $md(\Box F) = md(\Diamond F) = md(F) + 1$.

The semantic domain used to interpret modal logic formulas are Kripke structures. For later purposes it is advantagous to split the definition in two parts.

Definition 36 (Kripke Frame)
A **Kripke frame** (G, R) *consists of an arbitrary non-empty set* G *and an arbitrary binary relation* R *on* G.
Traditionally the elements of G *are called* **possible worlds** *and the relation* R *is called the* **accessability relation** .

Definition 37 (Propositional modal Kripke structure)
A **propositional Kripke structure** $\mathcal{K} = (G, R, v)$ *consists of a Kripke frame* (G, R) *together with a mapping* $v : G \times \mathrm{PVar} \to \{0, 1\}$, *which associates with every propositional variable* $p \in \mathrm{PVar}$ *depending on a possibly word* $g \in G$ *a truth value* $v(g, p)$.
If $v(g, p) =$ **true** *we say that* p *is true in the world* g.

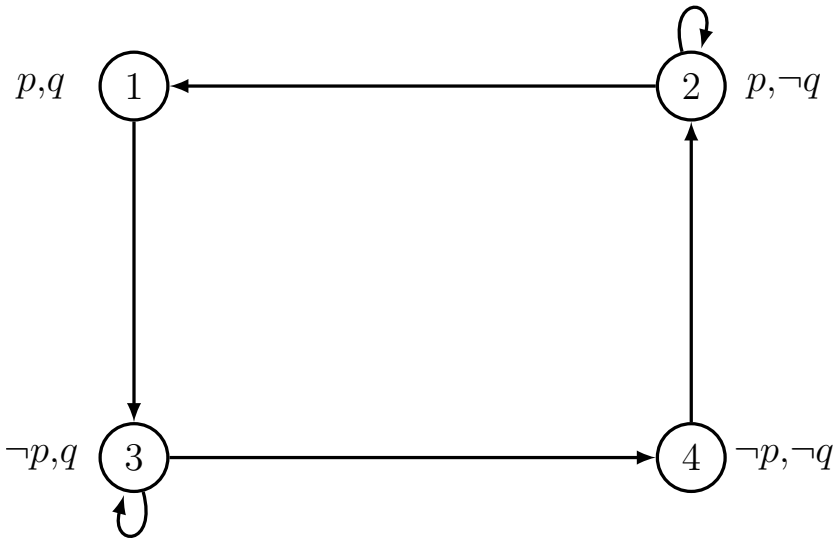


Figure 3.1: Example of a Kripke structure

If a Kripke structure contains not too many possible worlds – which will be the case for most examples we will encounter in this text – it is possible to represent it graphically. Figure 3.1 shows a Kripke structure with four worlds $G = \{1, 2, 3, 4\}$ and the accessability relation $R =$

$\{(1,3),(2,1),(2,2),(3,3),(3,4),(4,2)\}$. The function v can be read of to be $v(1,p) = v(2,p) = \mathbf{true}$, $v(3,p) = v(4,p) = \mathbf{false}$, $v(1,q) = v(3,q) = \mathbf{true}$ and $v(2,q) = v(4,q) = \mathbf{false}$.

Definition 38 *Let $\mathcal{K} = (G,R,v)$ be a Kripke structure, g a world in G and $F \in PModFml$.*
The relation $(\mathcal{K},g) \models F$, which we read as „ F is true (or valid) in world g of the Kripke structure $\mathcal{K}$“, is inductively defined as follows.
For simplicity of notation we write $g \models F$ instead of $(\mathcal{K},g) \models F$

1	$g \models p$	*iff*	$v(g,p) = \mathbf{true}$ *for* $p \in \mathrm{PVar}$
2	$g \models F \wedge H$	*iff*	$g \models F$ *and* $g \models H$
3	$g \models F \vee H$	*iff*	$g \models F$ *or* $g \models H$
4	$g \models F \to H$	*iff*	(*not* $g \models F$) *or* $g \models H$
5	$g \models \neg F$	*iff*	*not* $g \models F$
6	$g \models \Box F$	*iff*	*for all* h *such that* $R(g,h)$ *it is true that* $h \models F$
7	$g \models \Diamond F$	*iff*	*there exists a world* h *with* $R(g,h)$ *such that* $h \models F$

Definition 39 *A formula $F \in PModFml$ is called a modal tautology if for all Kripke structures $\mathcal{K} = (G,R,v)$ and all $g \in G$*

$$g \models F$$

is true.

Lemma 52 *The following formulas are modal tautologies.*

1. $\Box F \leftrightarrow \neg\Diamond\neg F$
2. $\Box(P \to Q) \to (\Box P \to \Box Q)$
3. $\Box(P \wedge Q) \leftrightarrow (\Box P \wedge \Box Q)$
4. $\Diamond(P \vee Q) \leftrightarrow (\Diamond P \vee \Diamond Q)$
5. $(\Box P \vee \Box Q) \to \Box(P \vee Q)$
6. $\Diamond(P \wedge Q) \to (\Diamond P \wedge \Diamond Q)$

Proofs

1. The definition of $g \models \Box F$ reads

   $$\text{for all } h \in G \text{ with } R(g,h) \text{ it holds } h \models F$$

   A simple reformulation of this is

   $$\text{there is no } h \in G \text{ with } R(g,h) \text{ and } h \models \neg F$$

   But, this is by definition $g \models \neg\Diamond\neg F$.

2. Let $\mathcal{K} = (G, R, v)$ be an arbitrary Kripke structure and $g \in G$ an arbitrary world. Assume $g \models \Box(P \to Q)$ with the aim of showing $g \models \Box P \to \Box Q$. We procced by assuming in addition $g \models \Box P$ and need to prove $g \models \Box Q$. To do so we consider an arbitrary world $h \in G$ satisfying $R(g,h)$. We have finished if we can show $h \models Q$. Using our second assumption, $g \models \Box P$, we know in any case $h \models P$. The first assumption, $g \models \Box(P \to Q)$, now yields $h \models P \to Q$ and thus $h \models Q$.

3. $$\begin{array}{lll} g \models \Box(P \wedge Q) & \text{iff} & \text{for all } h \text{ with } R(g,h)\ h \models (P \wedge Q) \\ & \text{iff} & \text{for all } h \text{ with } R(g,h)\ h \models P \text{ and } h \models Q \\ & \text{iff} & \text{for all } h \text{ with } R(g,h)\ h \models P \\ & & \text{and} \\ & & \text{for all } h \text{ with } R(g,h)\ h \models Q) \\ & \text{iff} & g \models (\Box P \wedge \Box Q) \end{array}$$

4. Easily follows from 3 and 1.

5. $$\begin{array}{lll} g \models (\Box P \vee \Box Q) & \text{iff} & \text{for all } h \text{ with } R(g,h)\ h \models P \\ & & \text{or} \\ & & \text{for all } h \text{ with } R(g,h)\ h \models Q \\ & \text{implies} & \text{for all } h \text{ with } R(g,h)\ h \models (P \vee Q) \\ & \text{iff} & \Box(P \vee Q) \end{array}$$

6. Easily follows from 5 and 1.

The power and beauty of modal logic has its roots in the sheer unbounded variations of definition 37 by requiring specific properties of the accessability relation R. Possible restrictions are to consider only Kripke structures with

reflexive accessability relation, or transitive R, or Kripke structures where R is a partial order relation. A rather queer terminology of classes of Kripke structures has evolved over time and resisted any attempts of a uniform standardisation. Here are some of the most popular classes.

Definition 40 (Classes of Kripke structure)

K	=	*the class of all Kripke structure*
T	=	*the class of all Kripke structure with R reflexive*
S4	=	*the class of all Kripke structure with R reflexive and transitive*
S5	=	*the class of all Kripke structure with R an equivalence relation*
K4	=	*the class of all Kripke structure with R transitive*
B	=	*the class of all Kripke structure with R transitive and symmetric*
D	=	*the class of all Kripke structure with R serial*
		i.e. for all $g \in G$ there is $h \in G$ with $R(g, h)$

Note, that what is really going on here is that classes of Kripke frames C_0 are defined, e.g., $C_T = \{(G, R) \mid R \text{ is transitive } \}$. Then the classes C of Kripke structures whose frame belongs to C_0 are obtained.

The notion of tautology can now be relativized to restricted classes of Kripke structures.

Definition 41 *Let C be a class of Kripke frames. A formula $F \in PModFml$ is called a modal C-tautology if for all Kripke structures $\mathcal{K} = (G, R, v)$ with (G, R) in C and all $g \in G$*

$$g \models F$$

is true.

The modal tautologies defined in Definition 39 thus conicide with the **K**-tautologies from Definition 41 .

Lemma 53 *The following formulas are* **T***-tautologies, i.e., valid in all reflexive Kripke structures.*

1. $\Box p \to p$

2. $p \to \Diamond p$
3. $\Box\Box p \to \Box p$
4. $\Box\Diamond p \to \Diamond p$
5. $\Box p \to \Diamond\Box p$
6. $\Diamond p \to \Diamond\Diamond p$

Proofs

1. Let g be an arbitrary world in a Kripke structure $\mathcal{K} = (G, R, v)$ with transitive R. Assume $g \models \Box p$. We want to show $g \models p$. From $g \models \Box p$ we obtain $h \models p$ for all $h \in G$ with $R(g, h)$. Since R is reflexive we certainly have $R(g, g)$ and the proof is finished.

2. Since $\Box p \to p$ is a **T**-tautology by the previous item $\Box\neg p \to \neg p$ is also a **T**-tautology. Using tautology 1 from Lemma 52 we get that also $\neg\Diamond p \to \neg p$ is a **T**-tautology. By propositional reasoning this formula is logically equivalent to $p \to \Diamond p$.

3. If a formula F is a C-tautology then any formula F' arising from F by uniformly substituting the propositional atom p by an arbitrary formula is also a C-tautology. If we replace p by $\Box p$ in $\Box p \to p$, which is a **T**-tautology by item 1 we get that also $\Box\Box p \to \Box p$ is a **T**-tautology.

4. Same argument as in the previous item with p being replaced by $\Diamond p$.

5. Follows from 4 by replacing p by $\neg p$ and the same equivalence transformations as used in item 2.

6. Follows from 3 by replacing p by $\neg p$ and the same equivalence transformations as used in item 2.

∎

It remains to introduce the last principal notion of modal logic, that of modal logical inference. In contrast to classical logic the more complicated semantics of modal logic offers more than one intuitive notion of logical inference, a local and a global version. Furthermore, as we have done with the notion

of tautologies, also logical inference can be relativized to specific classes of Kripke structures.

Definition 42 *Let $M \subseteq PModFml$ und $F \in PModFml$, C a class of Kripke structures.*

1. $M \models_L F$ *(F is a (local) logical consequence of M)*
   iff for every Kripke structure $\mathcal{K} = (G, R, v)$ and every $g \in G$
   $g \models M$ implies $g \models F$

2. $M \models_G F$ *(F is a global logical consequence of M)*
   iff for every Kripke structure $\mathcal{K} = (G, R, v)$
   if $g \models M$ for all $g \in G$ then $g \models F$ for all $g \in G$

3. $M \models_L^C F$ *(F is a (local) logical consequence of M)*
   iff for every Kripke structure $\mathcal{K} = (G, R, v)$ in C and every $g \in G$
   $g \models M$ implies $g \models F$

4. $M \models_G^C F$ *(F is a global logical consequence of M)*
   iff for every Kripke structure $\mathcal{K} = (G, R, v)$ in C
   if $g \models M$ for all $g \in G$ then $g \models F$ for all $g \in G$

If we use the term logical inference without qualification we mean local logical consequence. This will be our main concept, while global logical inference is treated as a variation.

The difference between local and global logical inference is clearly illustrated by the deduction theorem.

Theorem 54 *(Deduction Theorem) Let $F_1, F_2 \in PModFml$.*

$$F_1 \models_L F_2 \text{ iff } \models_L F_1 \to F_2$$

The proof is obvious.
On the other hand $p \models_G \Box p$ is certainly true while $\not\models_G p \to \Box p$, i.e. The deduction theorem does not hold for the global consequence relation. See exercise 3.9.14 on page 154 for more on this topic.

The observant reader might at this point expect a discussion of decidability issues for modal logic. Since Kripke structures can be infinite it is not clear

whether satisifiability is decidable, despite the restriction to propositional formulas. We will show that satisfiability of PDL is decidable, Theorem 103 on page 194. Then Exercise 4.7.9 on page 207 states that this entails decidability of modal satisfiability as a special case.

3.2 Correspondence Theory

Correspondence Theory has been and still is an important part of the theory of modal logic. To introduce its subject we start with a simple example. Consider a reflexive frame (G, R), i.e., a frame satisfying $(G, R) \models \forall x R(x, x)$ or in other words a frame whose accessability relation R is reflexive. No matter what interpretation v we use to obtain a Kripke structure $\mathcal{K} = (G, R, v)$ all worlds $g \in G$ will satisfy $(\mathcal{K}, g) \models \Box p \to p$. Here and in the following p, q will be used to denote propositional variables. Is the converse also true? By the converse we mean, that given an arbitrary frame (G, R) such that for all v and all $g \in G$ $(\mathcal{K}, g) \models \Box p \to p$ is true for $\mathcal{K} = (G, R, v)$, then R is reflexive. Assume that this is not true. Then there is a frame (G, R) satisfying the stated property, but R is not reflexive. Thus there is $g_0 \in G$ with $\neg R(g_0, g_0)$. We define an interpretation function v_0 by

$$v_0(g, p) = \begin{cases} 1 & \text{if } R(g_0, g) \\ 0 & \text{otherwise} \end{cases}$$

For $\mathcal{K}_0 = (G, R, v_0)$ obviously $(\mathcal{K}_0, g_0) \models \Box p$ is true. But, by choice of g_0 we also note that $(\mathcal{K}_0, g_0) \models \neg p$. Thus the converse of the above observation is in fact true.

Definition 43 *Let $\mathcal{G}$ be a class of frames and $F \in PModFml$.*
We say that F **characterizes** *class $\mathcal{G}$, if for any frame (G, R) the following it true:*

$(G, R) \in \mathcal{G}$
iff
$(G, R, v) \models F$ *for every* v

The fact we proved above can now be phrased as

Lemma 55
The class of reflexive frames is characterized by the formula $\Box p \to p$.

The goal of correspondence theory can now be described as the search for further characterizations results like Lemma 55, more general characterization theorems and the investigations into the underlying principles and limitations of these theorems.

We proceed by taking a look at some more characterization results. Figure 3.2 lists some frequently encountered properties of frames.

1	reflexive	$\forall x R(x,x)$
2	symmetric	$\forall x \forall y (R(x,y) \rightarrow R(y,x))$
3	serial	$\forall x \exists y R(x,y)$
4	transitive	$\forall x \forall y \forall z (R(x,y) \wedge R(y,z) \rightarrow R(x,z))$
5	Euklidian	$\forall x \forall y \forall z (R(x,y) \wedge R(x,z) \rightarrow R(y,z))$
6	weakly functional	$\forall x \forall y \forall z (R(x,y) \wedge R(x,z) \rightarrow y = z)$
7	functional	weakly functional + serial
8	dense	$\forall x \forall y (R(x,y) \rightarrow \exists z (R(x,z) \wedge R(z,y)))$
9	weakly connective	$\forall x \forall y \forall z (R(x,y) \wedge R(x,z) \rightarrow (R(y,z) \vee y = z \vee R(z,y))$
10	weakly oriented	$\forall x \forall y \forall z ((R(x,y) \wedge R(x,z)) \rightarrow \exists w (R(y,w) \wedge R(z,w))$
11	confluent	$\forall x \forall y (R(x,y) \rightarrow \exists z (R(x,z) \wedge R(y,z)))$

Figure 3.2: Some properties of frames

Lemma 56

The following is a list of defining properties of classes of frames and their characterizing formulas.

1	*reflexive*	$\Box p \rightarrow p$
2	*symmetric*	$p \rightarrow \Box \Diamond p$
3	*serial*	$\Box p \rightarrow \Diamond p$
4	*transitive*	$\Box p \rightarrow \Box \Box p$
5	*Euklidian*	$\Diamond p \rightarrow \Box \Diamond p$
6	*weakly functional*	$\Diamond p \rightarrow \Box p$
7	*functional*	$\Diamond p \leftrightarrow \Box p$
8	*dense*	$\Box \Box p \rightarrow \Box p$
9	*weakly connective*	$\Box((p \wedge \Box p) \rightarrow q) \vee \Box((q \wedge \Box q) \rightarrow p)$
10	*weakly oriented*	$\Diamond \Box p \rightarrow \Box \Diamond p$
11	*confluent*	$\Diamond \Box p \rightarrow \Diamond p$

Proofs Part 1 has already been dealt with in the text preceeding Lemma 55. We will give a proof for part 4. The proofs of the remaining cases will

be Exercise 3.9.4.

Part 4 Let (G, R) be a transitive frame, v an arbitrary interpretation and $g \in G$. Let $\mathcal{K}$ denote the Kripke structure (G, R, v). We assume $(\mathcal{K}, g) \models \Box p$ and aim to show $(\mathcal{K}, g) \models \Box\Box p$. For all worlds g_1, g_2 satisfying $R(g, g_1)$ and $R(g_1, g_2)$ we need to show $(\mathcal{K}, g_2) \models p$. By transitivity of R also $R(g, g_2)$ is true. Thus the assumption $(\mathcal{K}, g) \models \Box p$ immediately yields $(\mathcal{K}, g_2) \models p$.
For the proof of the second part of characterizability we assume that (G, R) is not transitive. Thus there are $g_0, g_1, g_2 \in G$ with $R(g_0, g_1)$, $R(g_1, g_2)$ and $\neg R(g_0, g_2)$. We define the interpretation function v_0 by

$$v_0(g, p) = \begin{cases} 1 & \text{if } R(g_0, g) \\ 0 & \text{otherwise} \end{cases}$$

For $\mathcal{K}_0 = (G, R, v_0)$ we immediately get $(\mathcal{K}_0, g_0) \models \Box p$. Since on the other hand $(\mathcal{K}_0, g_2) \models \neg p$ is true, we obtain $(\mathcal{K}_0, g_0) \not\models \Box\Box p$. Altogether we have shown that the formula $\Box p \to \Box\Box p$ is not valid in $\mathcal{K}_0$.

∎

Let us consider a slightly more general characterization result. We will consider properties that are more complicated than those from Figure 3.2.

Definition 44
We will use the following abbreviation:

$$R^n(x, y) = \exists u_1 \ldots \exists u_{n-1}(R(x, u_1) \wedge \ldots \wedge R(u_i, u_{i+1}) \wedge \ldots \wedge R(u_{n-1}, y)).$$

For $n = 1$ we stipulate $R^1(x, y) = R(x, y)$ and for $n = 0$ we set $R^0(x, y) = x \doteq y$.

$$\begin{aligned} C(m, n, j, k) \quad = \quad & \forall w_1 \forall w_2 \forall w_3((R^m(w_1, w_2) \wedge R^j(w_1, w_3)) \\ & \to \exists w_4(R^n(w_2, w_4) \wedge R^k(w_3, w_4))) \end{aligned}$$

If $(G, R) \models C(m, n, j, k)$ we say that frame (G, R) has the C property with parameters m, n, j, k. Figure 3.3 offers a visual explanation of the C properties. For some choices of the parameters the C property is equivalent to some of the familiar properties considered above, see Exercise 3.9.5.

Theorem 57
For every quintuple m, n, j, k of natural numbers $\Diamond^m\Box^n p \to \Box^j\Diamond^k p$ is a characterizing formula for the class of all frames satisfying $C(m, n, j, k)$.

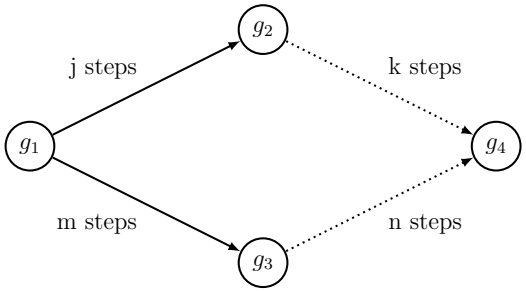


Figure 3.3: Visualization of the C property

Here we use $\Box^n p$ *as an abbreviation for* $\underbrace{\Box \ldots \Box}_{n-times} p$ *with* $\Box^0 p = p$ *and likewise for* $\Diamond^n p$.

Proof: We will throughout tacitly use Exercise 3.9.6.
For the first part we consider a frame (G, R) such that $(G, R) \models C(m, n, j, k)$. For every Kripke structure $\mathcal{K} = (G, R, v)$ and every $g \in G$ such that $(\mathcal{K}, g) \models \Diamond^m \Box^n p$ is true we need to show $(\mathcal{K}, g) \models \Box^j \Diamond^k p$. From $(\mathcal{K}, g) \models \Diamond^m \Box^n p$ we infer the existence of a world $g_1 \in G$ with $(G, R) \models R^m(g, g_1)$ and $(\mathcal{K}, g_1) \models \Box^n p$. To show $(\mathcal{K}, g) \models \Box^j \Diamond^k p$ we have to convince ourselves that for an arbitrary world g_2 satisfying $(G, R) \models R^j(g, g_2)$ also $(\mathcal{K}, g_2) \models \Diamond^k p$ is true. From the assumption $(G, R) \models C(m, n, j, k)$ we immediately obtain a world $h \in G$ with $(G, R) \models R^n(g_1, h)$ and $(G, R) \models R^k(g_2, h)$. In addition $(\mathcal{K}, g_1) \models \Box^n p$ yields $v(h, p) = 1$. By definition of $R^k(g_2, h)$ this implies $(\mathcal{K}, g_2) \models \Diamond^k p$.
For the second part of the proof we consider a frame (G, R), such that for all v we know $(G, R, v) \models \Diamond^m \Box^n p \to \Box^j \Diamond^k p$. To prove $(G, R) \models C(m, n, j, k)$ we pick three arbitrary worlds $g, g_1, g_2 \in G$ with the properties $(G, R) \models R^m(g, g_1)$ and $(G, R) \models R^j(g, g_2)$. The task is to find a *confluent* $h \in G$. To this end we define v by:

$$v(g, p) = \begin{cases} 1 & \text{if } R^n(g_1, g) \\ 0 & \text{otherwise} \end{cases}$$

For $\mathcal{K} = (G, R, v)$ this definition yields $(\mathcal{K}, g_1) \models \Box^n p$ as well as $(\mathcal{K}, g) \models \Diamond^m \Box^n p$. By assumption we know $(\mathcal{K}, g) \models \Box^j \Diamond^k p$ and thus also $(\mathcal{K}, g_2) \models \Diamond^k p$. There is thus $h \in G$ with $R^k(g_2, h)$ and $v(g_2, p) = 1$. By definition of v we must have $R^n(g_1, h)$.

Similar proofs may be found in [Sterling, 1992] and [Hughes & Cresswell, 1984]. A characterisation result for a much wider class of modal formulas is presented in [Sahlqvist, 1975]. A short presentation of Sahlqvist's theory is also contained in [Marx & Venema, 1997, Appendix B].

Before we go on and try to place the characterization problem into a wider context let us ask an obvious question. Do characterizing formulas always exist? This question is far to vague to be interesting. It is hopeless to search for characterizing formulas for every class $\mathcal{G}$ of frames. There are countably many formulas in $PModFml$ but uncountably many classes of frames. The frame classes we have seen so far were all of the form $\mathcal{G} = \{(G, R) \mid (G, R) \models \phi\}$ for a first-order formula ϕ. This suggest to replace the vague question above by the precise questions:

1. Given an arbitrary formula ϕ of first-order predicate logic is there a characterizing formula $F \in PModFml$ for the class $\mathcal{G} = \{(G, R) \mid (G, R) \models \phi\}$ of frames defined by ϕ?
2. Given an arbitrary formula $F \in PModFml$ let $\mathcal{G}$ be the class of frames (G, R) such that for all v the Kripke structure (G, R, v) satisfies F. Can $\mathcal{G}$ be defined by a first-order formula ϕ?

To cut a long story short, the answer to both questions is *no*. As a negative answer to the first question we will eventually show that there is no characterizing formula for the class of irreflexive frames.

It is a well known phenomenon that as a rule results claiming that an object with certain properties does **not** exists are far more involved than positive existence results. The case at hand is no exception. So we need some warmup before we set out to prove Lemma 59 below. The crucial construction in this proof is presented in the next definition.

Definition 45 *Let $\mathcal{K} = (G, R, v)$ be an arbitrary Kripke structure. For every $g \in G$ let g^1, g^2 be two new worlds different from each other and all worlds in G. A new Kripke structure $\mathcal{K}^* = (G^*, R^*, v^*)$ is defined by*

$$\begin{array}{lcll} G^* & = & \{g^i \mid g \in G, i \in \{1,2\}\} & \\ R^*(g^i, h^j) & \textit{iff} & R(g,h) \;\&\; i,j \in \{1,2\} & \textit{if } g \neq h \\ R^*(g^i, g^j) & \textit{iff} & R(g,g) \;\&\; i \neq j & \\ v^*(g^i, p) & = & v(g,p) & \end{array}$$

The important feature in this definition is that $R^*(g^1, g^1)$ is never true, neither is $R^*(g^2, g^2)$. In case the orginal relation R satisfies $\neg R(g, g)$ then none of the theoretically four possible relations involving R^* is true. In case $R(g, g)$ is true in the old Kripke structure then exactly $R^*(g^1, g^2)$ and $R^*(g^2, g^1)$ are true in the new Kripke structure.

Lemma 58
For every Kripke structure $\mathcal{K}$ and every world g of $\mathcal{K}$ the following is true for every formula $F \in PModFml$:

$$\begin{array}{lllll} 1 & (\mathcal{K}, g) \models F & \textit{iff} & (\mathcal{K}^*, g^1) \models F \\ 2 & (\mathcal{K}, g) \models F & \textit{iff} & (\mathcal{K}^*, g^2) \models F \\ 3 & \mathcal{K} \models F & \textit{iff} & \mathcal{K}^* \models F \end{array}$$

Proof
We will prove 1 and 2 by simultaneous induction on the complexity of F. Claim 3 is an immediate consequence of 1 and 2.
For the base case of the induction, $F = p$, (p a propositional atom) both claims are an immediate consequence of the definition of v^*.

The cases of the induction step where $F = F_1 \wedge F_2$, $F = \neg F_1$, or $F = F_1 \vee F_2$ are simple. We will present the details in the case $F = \Box F_1$. The only missing case, $F = \Diamond F_1$, can be handled completely analogously.

We assume $(\mathcal{K}, g) \models \Box F_1$ and want to arrive at $(\mathcal{K}^*, g^i) \models \Box F_1$. This assumption implies

$$(\mathcal{K}, h) \models F_1 \text{ for all } h \in G \text{ mit } R(g, h)$$

To show $(\mathcal{K}^*, g^i) \models \Box F_1$ we need to consider all $u \in G^*$ with $R^*(g^i, u)$. It is a good strategy to split the proof in two cases.
First Case: $u = h^j$ for some $h \in G$, $j \in \{1, 2\}$ with $g \neq h$ and $R(g, h)$.
By assumption $(\mathcal{K}, h) \models F_1$ is true and using the induction hypothesis we get from this $(\mathcal{K}^*, h^j) \models F_1$, as desired
Second Case: $u = g^j$ and $R(g, g)$.
By assumption we know $(\mathcal{K}, g) \models F_1$ and the induction hypothesis yields again $(\mathcal{K}^*, g^j) \models F_1$. Note, that we must have $i \neq j$ in the second case and that at this point of the argument it is crucial that we prove 1 and 2 simultaneously.
It still remains to prove the reverse implication of the lemma. So we assume,

$\forall x \neg R(x,x)$	Irreflexivity	
$\forall x \forall y R(x,y)$	Universality	see [Fitting, 1993]
$\exists x \exists y R(x,y)$	Nontriviality	see [Fitting, 1993]
$\forall x \exists y (R(x,y) \wedge R(y,y))$		see [van Benthem, 1984]

Figure 3.4: Examples of non-characterizable frame classes

e.g., $(\mathcal{K}^*, g^1) \models \Box F_1$ and set out to show $(\mathcal{K}, g) \models \Box F_1$. (The second case $(\mathcal{K}^*, g^2) \models \Box F_1$ can of course be treated completely analogously.) We need to show $(\mathcal{K}, h) \models F_1$ for every $h \in G$ with $R(g,h)$. By definition of R^* we obtain from $R(g,h)$ the relation $R^*(g^1, h^2)$ regardless of $g = h$ or $g \neq h$. Making use of the case assumption we arrive at $(\mathcal{K}^*, h^2) \models F_1$. The induction hypothesis allows us to conclude $(\mathcal{K}, h) \models F_1$.

▪

Lemma 59 *Let $\mathcal{G}$ be the class of all irreflexive frames, i.e., those frames satisfying $\forall x \neg R(x,x)$.*
There is no characterizing formula in $PModFml$ for $\mathcal{G}$.

Proof
By way of contradiction we assume the existence of a characterizing formula $F \in PModFml$ of class $\mathcal{G}$. We consider the very special frame (G_1, R_1) with $G_1 = \{g_0\}$ and $R_1(g_0, g_0)$. The Kripke structure (G^*, R^*) constructed in Definition 45 is certainly irreflexive. For an arbitrary v by choice of F we must have $(G^*, R^*, v^*) \models F$ and $(G, R, v) \not\models F$. This contradicts Lemma 58. Thus, a characterizing formula F cannot exists.

▪

Futher examples of non-characterizable classes of frames are listed in Figure 3.4. Still more results can be found in [van Benthem, 1984].

We return to the second question above *Can every class of frames characterized by a modal formula be defined by a first-order formula?* As already remarked the answer is negative. Here is the example that proves it.

Lemma 60
The formula

$$(\Box p \to \Box\Box p) \wedge (\Box(\Box p \to p) \to \Box p)$$

characterizes all transitive frames (G, R), such that the inverse relation R^{-1} is wellfounded, i.e., there is no infinite chain of elements $g_0, g_1, \ldots, g_n \ldots$ in G such that $R(g_i, g_{i+1})$ for all $0 \leq i$.

Proof: can be found e.g., in [van Benthem, 1984, pp 195ff]. For the convenience of the reader and since it is not too complicated we present our version here.

We already know that $(\Box p \rightarrow \Box\Box p)$ characterizes the class of transitive frames, so we will only be concered with the second conjunct.

claim of part 1 If (G, R) is transitive but there is a interpretation v such that $(G, R, v) \not\models (\Box(\Box p \rightarrow p) \rightarrow \Box p)$ then R^{-1} is not wellfounded.
We abbreviate (G, R, v) by $\mathcal{K}$. There is thus a world $g_1 \in G$ with

$$(\mathcal{K}, g_1) \models \Box(\Box p \rightarrow p) \tag{3.1}$$

but

$$(\mathcal{K}, g_1) \models \neg\Box p \tag{3.2}$$

Statement 3.2 implies the existence of $g_2 \in G$ with $R(g_1, g_2)$ and

$$(\mathcal{K}, g_2) \models \neg p \tag{3.3}$$

From 3.1 we also obtain

$$(\mathcal{K}, g_2) \models \Box p \rightarrow p \tag{3.4}$$

Together 3.4 and 3.3 yield

$$(\mathcal{K}, g_2) \models \neg\Box p \tag{3.5}$$

This in turn implies the existence of a world $g_3 \in G$ satisfying $R(g_2, g_3)$ and

$$(\mathcal{K}, g_3) \models \neg p \tag{3.6}$$

Since R was assumed to be transitive we also have $R(g_1, g_3)$. Thus 3.1 yields again

$$(\mathcal{K}, g_3) \models \Box p \rightarrow p \tag{3.7}$$

Now it is easy to see that we obtain in this way an infinite sequence $g_0, \ldots, g_i, \ldots$ with $R(g_i, g_{i+1})$ for all i.

claim of part 2 If (G, R) is a transitive frame such that for $(G, R, v) \models$

$(\Box(\Box p \to p) \to \Box p)$ is true for all v then R^{-1} is wellfounded.
In this case we assume that R^{-1} is not wellfounded and will arrive at a contradiction. There is thus a sequence $g_i \in G$ with $R(g_i, g_{i+1})$ for all $i \geq 1$. We define v by:

$$v(h,p) = \begin{cases} 0 & \text{iff } R(h,g_i) \text{ for some } i \\ 1 & \text{otherwise} \end{cases}$$

As usual we abbreviate (G, R, v) by $\mathcal{K}$. From $R(g_i, g_{i+1})$ we get by this definition $(\mathcal{K}, g_i) \models \neg p$ and $R(g_{i+1}, g_{i+2})$ further yields $(\mathcal{K}, g_i) \models \neg\Box p$ for all i.
In the end we want to show that $\mathcal{K} \models (\Box(\Box p \to p) \to \Box p)$ is not true. We thus need an $h^* \in G$ with $(\mathcal{K}, h^*) \models \Box(\Box p \to p$, but $(\mathcal{K}, h^*) \models \neg\Box p$. We will in fact show that $(\mathcal{K}, h) \models \Box p \to p$ is true for arbitrary h. Thus $(\mathcal{K}, g) \models \Box(\Box p \to p)$ is true for any g. Thus $h^* = g_i$ for any i will do.
So we consider an arbitrary $h \in G$. We distinguish two cases. In case 1 we assume that is no i with $R(h, g_i)$. By definition of v this gives $v(h,p) = 1$ i.e., $(\mathcal{K}, h) \models p$ and thus also $(\mathcal{K}, h) \models \Box p \to p$. In the second case $R(h, g_i)$ for some i. Because of $(\mathcal{K}, g_i) \models \neg p$ we also get $(\mathcal{K}, h) \models \neg\Box p$ which trivially yields $(\mathcal{K}, h) \models \Box p \to p$ as desired.

∎

Corollary 61 *Let $\mathcal{G}$ be the class of frames characterized by the formula from Lemma 60. There is no first-order formula ϕ such that*

$$\mathcal{G} = \{(G,R) \mid (G,R) \models \phi\}$$

Proof: Follows from the fact that the wellfoundedness property cannot be axiomatized in any logic satisfying the compactness property. For the convenience of the reader we repeat this easy argument here. Assume, for the sake of a contradiction, that there is a first-order formula ϕ with the property stated in the corrolary. Consider the following infinite set of formulas

$$\Gamma = \{\phi\} \cup \{R(c_i, c_{i+1}) \mid i \geq 0\}$$

Here the c_i are new constant symbols. Obviously Γ is not satisfiable, but every finite subset of Γ is. In first-order logic this is not possible.

∎

We will give another interesting example using multimodal logic. In the particular multimodal logic there will not only be one box-operator $\Box$, but

two of them $\Box_a$ and $\Box_b$. In the frames for multimodal logic there will be instead of one accessability relation R two accessability relations R_a and R_b. The example to follow will make use of the concept of transitive closure.

Definition 46
The **transitive closure** *of a binary relation R is the smallest relation R_t with the properties*

1. *R_t is transitive and reflexive*
2. *$R \subseteq R_t$*

Theorem 62
The conjunction of the following multimodal formulas

$\Box_a p \rightarrow (p \wedge \Box_a \Box_b p)$
$\Box_a(p \rightarrow \Box_b p) \rightarrow (p \rightarrow \Box_a p)$

characterizes the class of all multimodal frames (G, R_a, R_b), such that R_a is the transitive closure of R_b.

Proof: For the easy part assume (G, R_a, R_b) is a multimodal frame with R_a the transitive closure of R_b. We need to convince ourselves that for any interpretation v and any $g \in G$ for both formulas F $(\mathcal{K}, g) \models F$ is true, with $\mathcal{K} = (G, R_a, R_b, v)$. To show $(\mathcal{K}, g) \models \Box_a p \rightarrow (p \wedge \Box_a \Box_b p)$, we assume $(\mathcal{K}, g) \models \Box_a p$ and aim to show $(\mathcal{K}, g) \models p \wedge \Box_a \Box_b p$. Since R_a is reflexive we get immediately $(\mathcal{K}, g) \models p$. For the proof of the second conjunct we aim to show for arbitrary h_1, h with $R_a(g, h_1)$ and $R_b(h_1, h)$ that $(\mathcal{K}, h) \models p$. Since R_a is the transitive closure of R_b this implies $R_a(g, h)$ and the assumption $(\mathcal{K}, g) \models \Box_a p$ indeed yields $(\mathcal{K}, h) \models p$. To show that the second formula is satisfied we assume $(\mathcal{K}, g) \models \Box_a(p \rightarrow \Box_b p)$ and $(\mathcal{K}, g) \models p$ and aim to prove $(\mathcal{K}, g) \models \Box_a p$. That is to say, for every h with $R_a(g, h)$ we have to show $(\mathcal{K}, h) \models p$. Since R_a is the transitive closure of R_b there are $h_0, \ldots, h_n$ with $h_0 = g$, $h_n = h$ and $R_b(h_i, h_{i+1})$ for all $0 \leq i < n$. The proof now proceeds by induction on n. For $n = 0$ we have $g = h$ and we are done. For the induction step we have as a hypothesis $(\mathcal{K}, h_{n-1}) \models p$. Since by definition of the transitive closure we also have $R_a(g, h_{n-1})$ the assumption $(\mathcal{K}, g) \models \Box_a(p \rightarrow \Box_b p)$ tells us that $(\mathcal{K}, h_{n-1}) \models p \rightarrow \Box_b p$ is true. The

induction hypothesis can now be used to arrive at $(\mathcal{K}, h_{n-1}) \models \Box_b p$ and $R_b(h_{n-1}, h_n)$ finally yields $(\mathcal{K}, h) \models p$ as desired.

For the difficult part we fix a multimodal frame (G, R_a, R_b) such that for any interpretation v both formulas are satisfied in the Kripke structure $\mathcal{K} = (G, R_a, R_b, v)$. The proof that R_a is the transitive closure of R_b is split into two parts:

part 1: $R_a \subseteq \text{Tclosure}(R_b)$

Let g, h be worlds in G with $R_a(g, h)$. We define a interpretation v as follows:

$$v(w, p) = \begin{cases} 1 & \text{if } (g, w) \text{ in Tclosure}(R_b) \\ 0 & \text{otherwise} \end{cases}$$

For any $g \in G$ we know $(\mathcal{K}, g) \models \Box_a(p \to \Box_b p) \to (p \to \Box_a p)$, where $\mathcal{K}$ abbreviates (G, R_a, R_b, v). To be able to make use of this fact, we will first aim to show $(\mathcal{K}, g) \models \Box_a(p \to \Box_b p)$.

To this end we consider a world $w \in G$ with $R_a(g, w)$ and $(\mathcal{K}, w) \models p$ with the obligation to show $(\mathcal{K}, w) \models \Box_b p$. By definition $(\mathcal{K}, w) \models p$ is only possible if (g, w) is in the transitive closure of R_b. For any other world w' with $R_b(w, w')$ also (g, w') is in the transitive closure of R_b and by definition of v we get $(\mathcal{K}, w') \models p$. This shows $(\mathcal{K}, g) \models \Box_a(p \to \Box_b p)$ and the characterizing properties of the second formula implies $(\mathcal{K}, g) \models (p \to \Box_a p)$. Since (g, g) is in the transitive closure of any relation we get by definition of v in particular $(\mathcal{K}, g) \models p$ and thus $(\mathcal{K}, g) \models \Box_a p$. At this point we remember the pair (g, h) satisfying $R_a(g, h)$ that we started out with. From $(\mathcal{K}, g) \models \Box_a p$ we obtain $(\mathcal{K}, h) \models p$. Again using the definition of v we see that (g, h) is indeed in the transitive closure of R_b.

part 2: $\text{Tclosure}(R_b) \subseteq R_a$

We start with a pair of worlds (g, h) in the transitive closure of R_b and aim to arrive at $R_a(g, h)$. The following interpretation will be useful:

$$v(w, p) = \begin{cases} 1 & \text{iff } R_a(g, w) \\ 0 & \text{otherwise} \end{cases}$$

Again we abbreviate (G, R_a, R_b, v) simply by $\mathcal{K}$. By assumption we know $(\mathcal{K}, g) \models \Box_a p \to (p \wedge \Box_a \Box_b p)$. We have taken care in the definition of v that $(\mathcal{K}, g) \models \Box_a p$ is true. Thus we also have

$$(\mathcal{K}, g) \models (p \wedge \Box_a \Box_b p) \tag{3.8}$$

Since (g, h) is in the transitive closure of R_b there are worlds $g_0, g_1, \ldots\ldots, g_k$

with $g_0 = g$, $g_k = h$ and $R_b(g_i, g_{i+1})$ for all $0 \leq i < k$. We will show $(\mathcal{K}, g_i) \models p$ for all $0 \leq i \leq k$ by induction on i. For $i = 0$ we get $g_0 = g$ and the claim is part of 3.8. In the induction step we start with $(\mathcal{K}, g_i) \models p$. From this we obtain by the definition of v that $R_a(g, g_i)$, which by 3.8 yields $(\mathcal{K}, g_i) \models \Box_b p$ and thus, as desired, $(\mathcal{K}, g_{i+1}) \models p$ since $R_b(g_i, g_{i+1})$. In the end we have shown $(\mathcal{K}, g_k) \models p$, which is by $g_k = h$ also $(\mathcal{K}, h) \models p$. The definition of v now yields $R_a(g, h)$ and we are done.

∎

Is there a more systematic approach to find out which classes $\mathcal{G}$ of frames characterized by a modal formula F are first-order definable? An approach that might even assist in computing the first-order definition from F in case it exists? We will indeed present such an approach. The masterplan is to define the class $\mathcal{G}$ by a second-order formula ϕ, that is quite straight forward, and then investigate if ϕ may be equivalent to a first-order formula. That is the tricky part. Thus the problem in modal logic we wanted to solve in the first place is reduced to a much more general problem that is of interest in many other totally different contexts.

3.3 The Tree-Property

We start by stipulating some basic notation.

Definition 47 (Trees)

1. *A path in the Kripke frame (G, R) is a finite sequence $\langle g_0, \ldots, g_{n-1} \rangle$ of elements $g_i \in G$ such that $R(g_i, g_{i+1})$ for all $0 \leq i < n - 1$.*

2. *If $\sigma = \langle g_0, \ldots, g_{n-1} \rangle$ is a path we say more precisely that σ is a path from g_0 to g_{n-1}.*

3. *If $\sigma = \langle g_0, \ldots, g_n \rangle$ is a path in (G, R) we denote by $\ell(\sigma)$ the last element of σ, thus $\ell(\sigma) = g_n$.*

4. *By $\sigma_0 + g$ we denote the path obtained from σ_0 by adding g at the end, i.e., if $\sigma_0 = \langle g_0, \ldots, g_n \rangle$ the $\sigma_0 + g$ is $\langle g_0, \ldots, g_n, g \rangle$.*

5. *(G, R) is called a* tree *if for any two g_1, g_2 there is exactly one path either from g_1 to g_2 or from g_2 to g_1.*

6. (G, R) *is called a* rooted tree *if* (G, R) *is a tree and there is* $r \in G$ *such that for all* $g \in G$ *there is a path from* g. *The element* r *is then called the* root.

If (G, R) is a tree then for no $g \in G$ will $R(g, g)$ be true. Otherwise, $\langle g \rangle$, $\langle g, g \rangle$ and $\langle g, g, g \rangle$ would be different paths all leading from g to g.
Furthermore, $G_{nr} = \mathbb{Z}$ with $R_{nr}(z_1, z_2)$ iff $z_2 = z_1 + 1$ is an example of a tree that is not rooted.

Definition 48 (Bounded Morphism)
Let $\mathcal{K}_1 = (G_1, R_1, v_1)$, $\mathcal{K}_2 = (G_2, R_2, v_2)$ *be Kripke structures.*
A function $\rho : G_1 \to G_2$ *is called a* bounded morphism *if*

1. *for all* $g \in G_1$ *and propositional atoms* $p \quad v_1(g, p) = v_2(\rho(g), p)$
2. *for all* $g, h \in G_1 \quad R_1(g, h)$ *implies* $R_2(\rho(g), \rho(h))$
3. *for all* $g \in G_1$, *if there exists* $h' \in G_2$ *with* $R_2(\rho(g), h')$ *then there is* $h \in G_1$ *such that* $R_1(g, h)$ *and* $h' = \rho(h)$.

Lemma 63 *Let* ρ *be a bounded morphism from* $\mathcal{K}_1 = (G_1, R_1, v_1)$ *to* $\mathcal{K}_2 = (G_2, R_2, v_2)$, *and* $g \in G_1$. *Then for arbitrary modal formulas* F

$$(\mathcal{K}_1, g) \models F \quad \Leftrightarrow \quad (\mathcal{K}_2, \rho(g)) \models F$$

Proof The proof proceeds by structural induction on the complexity of the modal formula F. If F is a propositional atom the claim follows directly from item 1 in Definition 48. The propositional induction steps are obvious. So we assume that the claim is true for F_0 and proceed to prove it for $F = \Box F_0$.
$\Rightarrow$ Starting from the assumption $(\mathcal{K}_1, g) \models F$ we try to prove $(\mathcal{K}_2, \rho(g)) \models F$. To this end consider $h' \in G_2$ with $R_2(\rho(g), h')$ with the aim to show $(\mathcal{K}_2, h') \models F_0$. By item 3 in Definition 48 we obtain an $h \in G_1$ satisfying $\rho(h) = h'$ and $R_1(g, h)$. By case assumption we get $(\mathcal{K}_1, h) \models F_0$. The induction hypothesis yields $(\mathcal{K}_2, \rho(h)) \models F_0$, as desired.
$\Leftarrow$ Starting from the assumption $(\mathcal{K}_2, \rho(g)) \models F$ we try to prove $(\mathcal{K}_1, g) \models F$. To this end consider $h \in G_1$ with $R_1(g, h)$ with the aim to show $(\mathcal{K}_2, \rho(h)) \models F_0$. By item 2 in Definition 48 we obtain $R_2(\rho(g), \rho(h))$. By

case assumption we get $(\mathcal{K}_2, \rho(h)) \models F_0$. Again, induction hypothesis yields the desired result.

∎

Example 8 *Let $\mathcal{K}^* = (G^*, R^*, v^*)$ be the Kripke structure constructed from $\mathcal{K} = (G, R, v)$ in Definition 45 then ρ given by*

$$\rho(g^i) = g \text{ for } i \in \{1, 2\}$$

is a bounded morphism from $\mathcal{K}^*$ to $\mathcal{K}$. Lemma 58 is thus a special instance of Lemma 63.

Example 9 *Let (G_2, R_2) be a closed subframe of (G_1, R_1), see Definition 90 and $\mathcal{K}_2 = (G_2, R_2, v_2)$ the restriction of $\mathcal{K}_1 = (G_1, R_1, v_1)$. Then the identity map $\rho_{id} : G_2 \to G_2$ is a bounded morphism from $\mathcal{K}_2$ in $\mathcal{K}_1$. The solution to Exercise 3.9.1 may then be seen as a special case of Lemma 63.*

Definition 49 (Unfolding) *Let $\mathcal{K} = (G, R, v)$ be a Kripke structure. For an element $g \in G$ the* unfolding $\vec{\mathcal{K}}_g = (\vec{G}_g, \vec{R}, \vec{v})$ *of $\mathcal{K}$ at g is given by*

$$\begin{array}{lcl} \vec{G}_g & = & \{\sigma \mid \sigma \text{ is non-empty path in } (G, R) \text{ starting with } g\} \\ \vec{R}(\sigma_1, \sigma_2) & \Leftrightarrow & \sigma_2 = \sigma_1 + \ell(\sigma_2) \\ & & \text{i.e., } \sigma_2 \text{ is an extension of } \sigma_1 \text{ by exactly one entry} \\ \vec{v}(\sigma, p) & = & v(\ell(\sigma), p) \end{array}$$

If the frame (G, R) contains a loop then $\vec{G}$ will be infinite.

Lemma 64 *Let $\mathcal{K} = (G, R, v)$ be a Kripke structure, $\vec{\mathcal{K}}_g = (\vec{G}_g, \vec{R}, \vec{v})$ its unfolding at g. Then*

$$(\mathcal{K}, g) \models F \quad \Leftrightarrow \quad (\vec{\mathcal{K}}_g, \langle g \rangle) \models F$$

for every modal formula F.

Proof Because of Lemma 63 it suffices to show that ρ given by $\rho(\sigma) = \ell(\sigma)$ is a bounded morphism from $\vec{\mathcal{K}}$ onto $\mathcal{K}$. Properties 1 and 2 of Definition 48 follow directly from the definition of $\vec{v}$ and $\vec{R}$. To check property 3 of Definition 48 we consider $\sigma \in \vec{G}_g$, $\sigma = \langle g_0, \ldots g_n \rangle$, and $h \in G$ with $R(\rho(\sigma), h)$. For $\sigma_1 = \langle g_0, \ldots g_n, h \rangle = \sigma + h$ we obtain $\rho(\sigma_1) = \ell(\sigma_1) = h$ and $\vec{R}(\sigma, \sigma_1)$.

▪

Corollary 65 *If a modal formula F is satisfiable, then there is a Kripke structure $\mathcal{K} = (G, R, v)$ with (G, R) a tree with root r such that*

$$(\mathcal{K}, r) \models F$$

Proof In view of Lemma 64 it suffices to convince ourselves that the unfolding at g $\vec{\mathcal{K}}_g = (\vec{G}_g, \vec{R}, \vec{v})$ of any Kripke structure $\mathcal{K}$ is a tree with root $\langle g \rangle$. Since we have for any path $\sigma_0, \ldots, \sigma_{n-1}$ in $(\vec{G}_g, \vec{R})$ that σ_0 must be an inital segment of σ_{n-1} this is obvious.

▪

3.4 Second Order Logic

Definition 50
The signature Σ of a logic consists of the two sets $FSym$ of function symbols and $PSym$ of predicate symbols.
There is furthermore a function $\alpha : FSym \cup PSym \to \mathbb{N}$ such that $\alpha(f)$ fixes the number of arguments of the function symbol f, respectively $\alpha(r)$ the number of arguments of the relation symbol r.

A function symbol f with $\alpha(f) = 0$ is called a constant symbol, a relation symbol r with $\alpha(r) = 0$ is a propositional variable. There is no difference between a signature for a first-order language and a signature for a second-order language. This statement is at least true for the second order language we consider here. For typed second-order languages one would e.g., have to adapt α.

The other syntactial elements that are used to built formulas, such as the propositional operators $\neg, \wedge, \vee, \to, \leftrightarrow$ and the quantifiers $\forall, \exists$ and auxiliary symbols like $(,)$ are called *logical symbols* and not listed in Σ. We will also consider the equality symbol $\doteq$ as a logical symbol. The last missing building blocks for formulas are variable symbols and this is where the differences between first and second order logic lie.

We still have symbols for first-order variables which we will usually denote by lower case letters from the end of the alphabet, $v_0, v_1, \ldots$ or $x, y, z, \ldots$.

A new feature are second order variables for unary, binary, n-ary relation symbols. These we will denote by upper case letters from the end of the alphabet with superscripts indicating the arity $V_0^1, V_1^1, \ldots$ or $X^2, Y^k, Z^3, \ldots$. Thus X^1 will be interpreted as a variable ranging over sets, X^2 as a variable ranging over binary relations etc. We will not quantify over 0-ary relations, which is no loss. Also in our version of second-order logic there will be no variables and thus no quantifiers ranging over functions. This is again no real loss since n-ary functions can be coded as $n+1$-ary relations.

Definition 51
Let Σ be a signature. The set of terms Term *is defined as usual:*

1. *every first-order variable x is a term.*

2. *if* $t_1, \ldots, t_n$ *are terms and* f *is an* n*-ary functions symbols in* $FSym$ *then* $f(t_1, \ldots, t_n)$ *is also a term.*

There is no difference between first-order or second-order terms.
The set of second-order formulas $FmlSO$ *are inductively defined by:*

1. $t_1 \doteq t_2$ *is an (atomic) formula in* $FmlSO$ *for any two terms* t_1, t_2,
2. $p(t_1, \ldots, t_n)$ *is an (atomic) formula in* $FmlSO$ *for terms* $t_1, \ldots, t_n$ *and any* n*-ary symbol* $p \in PSym$,
3. $X^n(t_1, \ldots, t_n)$ *is an (atomic) formula in* $FmlSO$ *for terms* $t_1, \ldots, t_n$ *and any* n*-ary second-order variable symbol* X^n,
4. *if* F_1, F_2 *are formulas in* $FmlSO$ *so are* $\neg F_1$, $F_1 \wedge F_2$ $F_1 \vee F_2$ $F_1 \rightarrow F_2$,
5. *if* F *is a formula in* $FmlSO$ *so are* $\forall x F$, $\exists x F$, $\forall X^n F$ $\exists X^n F$ *for variables* x, X^n.

If need arises we will write Term_Σ and $FmlSO_\Sigma$ instead of Term and $FmlSO$. We will frequently write X instead of X^1.

Formulas that do not contain second-order variables X^i with $i > 1$ are called formulas of *monadic second order*.

The notion of free and bound occurences of first and second-order variable follows the same pattern and we assume that no further details need be provided.

Definition 52
A structure $\mathcal{M} = (D, I)$ *for signature* $\Sigma = FSym \cup PSym$ *consists of a non-empty universe* D *and an interpretation function* I *such that*

1. *for every symbol* $f \in FSym$ *the value* $I(f)$ *is a function from* $D^{\alpha(f)}$ *into* D
2. *for every symbol* $r \in PSym$ *the value* $I(r)$ *is a subset of* $D^{\alpha(r)}$, *i.e., an* n*-ary relation on* D *for* $n = \alpha(r)$.

There is no difference between a first-order or second-order structure for Σ.

The crucial definition is the evaluation of a formula $F \in FmlSO$ for a given structure.

Definition 53
Let Σ be a signature, $\mathcal{M} = (D, I)$ a Σ-structure, β an assignment for first-order variables and γ an assignment for second-order variables, i.e.,

$\beta(x) \in D$
$\gamma(X^n) \subseteq D^n$

We will employ the following usual notation

$$\beta_x^d(y) = \begin{cases} \beta(y) & \text{if } x \neq y \\ d & \text{if } x = y \end{cases} \qquad \gamma_{X^n}^M(Y) = \begin{cases} \gamma(Y) & \text{if } X^n \neq Y \\ M & \text{if } X^n = Y \end{cases}$$

here $d \in D$ and $M \subseteq D^n$.

The evaluation $\mathrm{val}_{\mathcal{M},\beta,\gamma}(t)$ for arbitrary terms $t \in \mathrm{Term}$ is the same as in first-order logic and we skip it here. In this case γ has of course no influence on the result.
The evaluation $\mathrm{val}_{\mathcal{M},\beta,\gamma}(F)$ for arbitrary formulas $F \in FmlSO$ is inductively defined by:

$\mathrm{val}_{\mathcal{M},\beta,\gamma}(t_1 \doteq t_2) = 1$	*iff*	$\mathrm{val}_{\mathcal{M},\beta,\gamma}(t_1) = \mathrm{val}_{\mathcal{M},\beta,\gamma}(t_2)$
$\mathrm{val}_{\mathcal{M},\beta,\gamma}(p(t_1,\ldots,t_n)) = 1$	*iff*	$(\mathrm{val}_{\mathcal{M},\beta,\gamma}(t_1),\ldots,\mathrm{val}_{\mathcal{M},\beta,\gamma}(t_n)) \in I(p)$
$\mathrm{val}_{\mathcal{M},\beta,\gamma}(X^n(t_1,\ldots,t_n)) = 1$	*iff*	$(\mathrm{val}_{\mathcal{M},\beta,\gamma}(t_1),\ldots,\mathrm{val}_{\mathcal{M},\beta,\gamma}(t_n)) \in \gamma(X^n)$
$\mathrm{val}_{\mathcal{M},\beta,\gamma}(\exists x F) = 1$	*iff*	*there is $d \in D$ with* $\mathrm{val}_{\mathcal{M},\beta_x^d,\gamma}(F) = 1$
$\mathrm{val}_{\mathcal{M},\beta,\gamma}(\forall x F) = 1$	*iff*	*for all $d \in D$* $\mathrm{val}_{\mathcal{M},\beta_x^d,\gamma}(F) = 1$
$\mathrm{val}_{\mathcal{M},\beta,\gamma}(\exists X^n F) = 1$	*iff*	*there is $M \subseteq D^n$ with* $\mathrm{val}_{\mathcal{M},\beta,\gamma_{X^n}^M}(F) = 1$
$\mathrm{val}_{\mathcal{M},\beta,\gamma}(\forall X^n F) = 1$	*iff*	*for all $M \subseteq D^n$* $\mathrm{val}_{\mathcal{M},\beta,\gamma_{X^n}^M}(F) = 1$

all remaining cases are as in first-order logic

If F contains no free variables we write $val_{\mathcal{M}}(F)$ instead of $\mathrm{val}_{\mathcal{M},\beta,\gamma}(F)$ and also use $\mathcal{M} \models F$ in place of $val_{\mathcal{M}}(F) = 1$.

Example 10 *Here is a simple example with empty Σ. For the formula $E(x,y) \equiv \forall X(X(x) \leftrightarrow X(y))$ we get $\mathrm{val}_{\mathcal{M},\beta,\gamma}(E) = 1$ iff $\beta(x) = \beta(y)$. In other words $E(x,y)$ defines equality.*

Example 11 *Let $\Sigma = FSym \cup PSym$ with $FSym = \emptyset$ and $PSym = \{R\}$ with R a binary relation symbol, that is, Σ is the signature for Kripke*

frames. Let $\mathcal{G}_{refl}$ *be the class of frames characterized by* $\Box p \to p$ *and* F_{refl} *the formula* $\forall X(\forall x(\forall y(R(x,y) \to X(y)) \to X(x))$.

$$
\begin{array}{lll}
(G,R) \in \mathcal{G}_{refl} & \textit{iff} & \textit{for all } v \textit{ we have } (G,R,v) \models \Box p \to p \\
& \textit{iff} & \textit{for all } v \textit{ and all } g \in G \textit{ we have } ((G,R,v),g) \models \Box p \to p \\
& \textit{iff} & \textit{for all } v \textit{ and all } g \in G \textit{ we have} \\
& & \quad \textit{if } ((G,R,v),g) \models \Box p \\
& & \quad \textit{then } ((G,R,v),g) \models p \\
& \textit{iff} & \textit{for all } v \textit{ and all } g \in G \textit{ we have} \\
& & \quad \textit{if } ((G,R,v),h) \models p \textit{ for all } h \textit{ with } R(g,h) \\
& & \quad \textit{then } ((G,R,v),g) \models p \\
& \textit{iff} & \textit{for all } v \textit{ and all } g \in G \textit{ we have} \\
& & \quad \textit{if } v(p,h) = 1 \textit{ for all } h \textit{ with } R(g,h) \\
& & \quad \textit{then } v(p,g) = 1 \\
& \textit{iff} & (G,R) \models F_{refl}
\end{array}
$$

The last equivalence depends on the fact that the interpretation v *and the assigment* $\gamma(X)$ *carry the same information. Given* v *we can define* $\gamma(X) = \{g \in G \mid v(p,g) = 1\}$ *and if* $\gamma(X) = M$ *is given we set* $v(p,g) = 1 \Leftrightarrow g \in M$.

Example 11 showed an example of a second-order formula that axiomatized exactly those frames that are characterized by the modal formula $\Box p \to p$. This is just one example of the general phenomenon stated in the next lemma.

Lemma 66 *For every modal formula* $F \in PModFml$ *there is a formula* $F^* \in FmlSO$ *such that for all frames* (G,R)

$$(G,R) \models F^* \Leftrightarrow (G,R,v) \models F \textit{ for all } v$$

Proof As it stands the claim cannot be proved via induction on the complexity of F. We need to allow formulas F^* with free first and second order variables. For every propositional variable p occuring in F let X_p be a second-order variable. F^2 will contain exactly one free first-order variable, which we will always denote by x.

The claim now reads:
For every modal formula F there exist a second-order formula F^2 with x as

its only free first-order variable and at most the free second-order variable X_p for the propositional atoms p in F such that:

$$\mathrm{val}_{(\mathrm{G,R}),\beta,\gamma}(F^2) = 1 \Leftrightarrow ((G,R,v),g) \models F \tag{3.9}$$

with $\beta(x) = g$ and $\gamma(X_p) = \{h \in G \mid v(p,h) = 1\}$.
Next we show how F^2 can be explicitely constructed from F. We will denote by $G(u/w)$ the formula that arises from G by replacing all free occurences of the variable w by u.

F	F^2
p	$X_p(x)$
$F_1 \wedge F_2$	$F_1^2 \wedge F_2^2$
$F_1 \vee F_2$	$F_1^2 \vee F_2^2$
$\neg F_0$	$\neg F_0^2$
$\Box F_0$	$\forall y(R(x,y) \rightarrow F_0^2(y/x))$
$\Diamond F_0$	$\exists y(R(x,y) \wedge F_0^2(y/x))$

F^* is the formula $\forall x \forall X_{p_1} \ldots \forall X_{p_k} F^2$ where $p_1, \ldots, p_k$ are all propositional variables in the modal formula F.

It is now a matter of routine to verify inductively that this construction yields formulas F^2 that satisfy 3.9. We present the details for the base case $F = p$ and the induction step $F = \Box F_1$.
$\mathbf{F = p}$ In this case $F^2 = X_p(x)$.

$$\begin{aligned} \mathrm{val}_{(\mathrm{G,R}),\beta,\gamma}(X_p(x)) = 1 &\Leftrightarrow \beta(x) \in \gamma(X_p) \\ &\Leftrightarrow g \in \{h \in G \mid v(p,h) = 1\} \\ &\Leftrightarrow v(p,g) = 1 \\ &\Leftrightarrow ((G,R,v),g) \models p \end{aligned}$$

$\mathbf{F = \Box F_0}$ In this case $F^2 = \forall y(R(x,y) \rightarrow F_0^2(y/x))$.

$$\begin{aligned} \mathrm{val}_{(\mathrm{G,R}),\beta,\gamma}(F^2) &\Leftrightarrow \mathrm{val}_{(\mathrm{G,R}),\beta',\gamma}(F_0^2) \text{ for all } h \in G \text{ with } R(g,h) \\ &\quad \text{and } \beta' = \beta_x^h. \\ &\Leftrightarrow ((G,R,v),h) \models F_0 \text{ for all } h \in G \text{ with } R(g,h) \\ &\quad (\textit{ind.hyp.}) \\ &\Leftrightarrow ((G,R,v),g) \models \Box F_0 \end{aligned}$$

■

Let us now return to Example 11. It can been seen without too much difficulty, using basically the same type of arguments as in the proof of Lemma

55, that $\forall X(\forall x(\forall y(R(x,y) \rightarrow X(y)) \rightarrow X(x))$ is logically equivalent to $\forall x R(x,x)$. An equivalence of this type is called an *elimination of second-order quantifiers*. Is there a more systematic way to eliminate second-order quantifiers? Or put in different words, is there a more systematic way to find for a second-order formula an equivalent first-order formula? Let us hasten to say that most second-order formulas do not have an equivalent first-order formula and the question of deciding if a second-order formulas has an equivalent first-order formula is undecidable, see e.g., [M.Gabbay *et al.*, 2008, Subsection 3.4.6]. But, there are interesting subclasses of formulas that allow second-order quantifier elimination. The basis result, presented in the next lemma, has first been proved by Ackermann in [Ackermann, 1935]. We need one auxiliary definition.

Definition 54 *Let F be a second-order formula, F_0 a subformula of F.*

1. *an occurence of F_0 as a subformula of F is called* positive *(* negative*) if it occurs in the scope of an even (odd) number of negation symbols. Counting scopes of negation symbols is done after transformation into a normal form where $\neg$,$\wedge$ and $\vee$ are the only propositional connectives, or occurences within the left-hand side of an implication are counted as well.*

2. *Let R be a k-ary relation symbol. F is called positive (negative) for R if all occurences of formulas $R(\bar{t})$ in F are positive (negative).*

3. *Let X^k be a k-ary second-order variable. F is called positive (negative) for X^k if all occurences of formulas $X^k(\bar{t})$ in F are positive (negative).*

Lemma 67 *Let F be a second-order formula, with X^k as its only free second-order variable. Assume that* $\mathrm{val}_{\mathcal{M},\beta,\gamma}(F) = 1$.

1. *if X^k is positive in F then also* $\mathrm{val}_{\mathcal{M},,\beta,\gamma'}(F) = 1$ *for all γ' with $\gamma(X) \subseteq \gamma'(X)$.*

2. *if X^k is negative in F then also* $\mathrm{val}_{\mathcal{M},,\beta,\gamma'}(F) = 1$ *for all γ' with $\gamma'(X) \subseteq \gamma(X)$.*

Proof Easy induction and also pretty obvious.

∎

Lemma 68 (Ackermann's Lemma) *Let X^k be a second-order variable, and $\bar{x}$ a k-tuple of first-order variables. Let G be a first-order formula and H a formula without second-order quantifiers and at most X^k as a free second-order variable.*

1. *If H is positive for X^k then*

$$\exists X^k(\forall \bar{x}(X^k(\bar{x}) \to G) \wedge H) \leftrightarrow H^*$$

2. *If H is negative for X^k then*

$$\exists X^k(\forall \bar{x}(G \to X^k(\bar{x})) \wedge H) \leftrightarrow H^*$$

In both cases H^* is obtained from H by replacing any atomic second-order subformula $X^k(\bar{t})$ by G. More precisely, we write $X^k(\bar{t})$ as $X^k(\bar{x})[\bar{t}/\bar{x}]$, which is always possible, and substitute it by $G[\bar{t}/\bar{x}]$.
Here the notation $G[t_1, \ldots, t_k/x_1, \ldots, x_k]$ stands for the formula that arises from G by the simultaneous substitution of t_i for x_i for all $1 \leq i \leq k$.

Proof We will only show the details of part 1 of the lemma. This should give the reader the right ideas how to proof part 2.

$\to$ By assumption we know for an arbitrary structure $\mathcal{M} = (D, I)$ and arbitrary valuations β,γ $\mathrm{val}_{\mathcal{M},\beta,\gamma}(\exists X^k(\forall \bar{x}(X^k(\bar{x}) \to G) \wedge H) = 1$. There exists a subset $V \subseteq D^k$ such that $\mathrm{val}_{\mathcal{M},\beta,\gamma_0}(\forall \bar{x}(X^k(\bar{x}) \to G)) = 1$ and $\mathrm{val}_{\mathcal{M},\beta,\gamma_0}(H) = 1$ with $\gamma_0 = \gamma_X^V$. Let W be the set $\{\bar{d} \in D^k \mid val_{\mathcal{M},\beta\frac{\bar{d}}{\bar{x}}}(G) = 1\}$. From $\mathrm{val}_{\mathcal{M},\beta,\gamma_0}(\forall \bar{x}(X(\bar{x}) \to G)) = 1$ we get $V \subseteq W$ and from the positivity of H we get $\mathrm{val}_{\mathcal{M},\beta,\gamma_1}(H) = 1$ with $\gamma_1 = \gamma_X^W$. To obtain H^* a subformula $X^k(\bar{x})[\bar{t}/\bar{x}]$ of H is replaced by $G[\bar{t}/\bar{x}]$. Now, $\mathrm{val}_{\mathcal{M},\beta,\gamma_1}(X^k(\bar{x})[\bar{t}/\bar{x}]) = 1$ iff $val_{\mathcal{M},\beta}(\bar{t}) \in W$ iff $val_{\mathcal{M},\beta}(G[\bar{t}/\bar{x}]) = 1$. This leads us to $\mathrm{val}_{\mathcal{M},\beta,\gamma_1}(H^*) = 1$.

Finally, since H^* is a purely first-order formula this is the same as $\mathrm{val}_{\mathcal{M},\beta,\gamma}(H^*) = 1$ or $val_{\mathcal{M},\beta}(H^*) = 1$.

$\leftarrow$ We start with $\mathrm{val}_{\mathcal{M},\beta,\gamma}(H^*) = 1$. Let, as above, W be the set $\{\bar{d} \in D^k \mid val_{\mathcal{M},\beta\frac{\bar{d}}{\bar{x}}}(G) = 1\}$. By the same argument as in the first part

of the proof we obtain $\mathrm{val}_{\mathcal{M},\beta,\gamma^{\mathrm{w}}_{\mathrm{X}}}(H) = 1$. By choice of W we also have $\mathrm{val}_{\mathcal{M},\beta,\gamma^{\mathrm{w}}_{\mathrm{X}}}(\forall\bar{x}(X^k(\bar{x}) \to G)) = 1$. This shows $\mathrm{val}_{\mathcal{M},\beta,\gamma}(\exists X^k(\forall\bar{x}(X^k(\bar{x}) \to G) \wedge H)) = 1$ as desired.

∎

Example 12 *Let us try to apply Lemma 68 to the formula F_{refl}. We write $F_{refl} \equiv \forall x F_0$ with F_0 : (rename variables y to x and x to z to avoid notational conflicts with the lemma.)*

$$\forall X(\forall x(R(z,x) \to X(x)) \to X(z))$$

from Example 11. Since the lemma only talks about existential second-order quantifiers we consider the negated formula $\neg F_0$:

$$\exists X(\forall x(R(z,x) \to X(x)) \wedge \neg X(z))$$

This formula satisfies the assumptions of the lemma with $G \equiv R(z,x)$ and $H \equiv \neg X(z)$. Since H is obviously negative for X we are in case 2 of the lemma. H^* is computed as $\neg R(z,z)$. Thus $\neg F_0 \leftrightarrow \neg R(z,z)$, i.e., $F_0 \leftrightarrow R(z,z)$ and thus also $F_{refl} \leftrightarrow \forall z F_0$ $(\equiv \forall z R(z,z))$.

Let us look at another example that shows that the application of Lemma 68 is not always so straight forward.

Example 13 *We start with the modal formula $p \to \Box\Diamond p$ which we know characterizes the class of symmetric frames. Translation into second-order logic following the construction from Lemma 66 and subsequent negation yields.*

1 F_{sym} $\forall X(X(z) \to \forall u(R(z,u) \to \exists x(R(u,x) \wedge X(x))))$
2 $\neg F_{sym}$ $\exists X(X(z) \wedge \exists u(R(z,u) \wedge \forall x(R(u,x) \to \neg X(x))))$

To make this formula fit the pattern of Ackermann's lemma we move the quantifier $\exists u$ in front of the formula, which can easily seen to be possible. From now on u will be a free variable just as z. This leads to

3 $\neg F'_{sym}$ $\exists u \exists X(X(z) \wedge R(z,u) \wedge \forall x(R(u,x) \to \neg X(x)))$
or equivalently
4 $\neg F''_{sym}$ $\exists X(\forall x(X(x) \to \neg R(u,x)) \wedge X(z) \wedge R(z,u))$

Now we can apply Ackermann's lemma with $G \equiv \neg R(u,x)$ and

$H \equiv X(z) \wedge R(z,u)$.

5	$\neg F''_{sym}$	$\leftrightarrow$	$\neg R(u,z) \wedge R(z,u)$
more precisely			
5*a*	$\forall u(\neg F''_{sym}$	$\leftrightarrow$	$\neg R(u,z) \wedge R(z,u))$
which leads to			
6	$\neg F'_{sym}$	$\leftrightarrow$	$\exists u(\neg R(u,z) \wedge R(z,u))$
reversing the prenexing step for $\exists u$			
6*a*	$\neg F_{sym}$	$\leftrightarrow$	$\exists u(\neg R(u,z) \wedge R(z,u))$
and finally to			
7	F_{sym}	$\leftrightarrow$	$\forall u(R(z,u) \rightarrow R(u,z))$

See Exercises 3.9.10 and 3.9.11 for more examples.

Algorithms have been proposed that automate the elimination of second-order quantifiers, e.g., the DLS algorithm [Doherty *et al.*, 1997] that has e.g., been implemented in [Gustafsson, 1996], Additional information can be found in [M.Gabbay *et al.*, 2008].

In [Conradie *et al.*, 2009, Conradie *et al.*, 2006b, Conradie *et al.*, 2006a] an algorithm for computing first-order equivalents in modal logic was introduced, called SQEMA, which is based on a modal version of the Ackermann Lemma. The algorithm can be accessed online at `http://www.fmi.uni-sofia.bg/fmi/logic/sqema/sqema.jsp`

3.5 A Tableau Calculus

We assume that the reader is familiar with the basic concepts of the tableau calculus for first-order logic, as it is e.g., contained in [Fitting, 1990].

In this section we present a tableau calculus for propositional modal logic following [Fitting, 1983]. Since the aspects of first-order logic and modal operators do not interfere it is not difficult to extend this calculus to first-order modal logic.

Definition 55 (Prefix)

1. *A* prefix σ *is a finite sequence of natural numbers.*
2. *A* prefix formula *is a syntactical element of the form* σA *where* σ *is a prefix and* A *a modal formula in* $PModFml$
3. *A* signed prefix formula *is a syntactical element of the form* $\sigma Z A$*, with* σA *a prefix formula and* $Z \in \{T, F\}$*.*

Thus $\langle 11\rangle((p \wedge q) \rightarrow (\Box p \vee \Diamond q))$ is a prefix formula. Note, that $\langle 11\rangle(p \wedge q) \rightarrow \langle 1\rangle(\Box p \vee \Diamond q)$ is not a prefix formula. A prefix can only occur at the top level. In examples we will write prefixes as $\langle n_1 n_2 \ldots n_k\rangle$ instead of $\langle n_1, n_2, \ldots, n_k\rangle$. This is possible since we will only need to consider prefixes with single digit n_i. Thus $\langle 111\rangle$ will stand for $\langle 1, 1, 1\rangle$ and not for $\langle 1, 11\rangle, \langle 11, 1\rangle$ or $\langle 111\rangle$. By σn we will denote the string that arises from σ by adding the single digit n.

It remains to explain when a signed formula is true at a state g in a Kripke structure (G, R, v). But, this is obvious:

$$g \models TA \quad \Leftrightarrow \quad g \models A$$
$$g \models FA \quad \Leftrightarrow \quad g \models \neg A$$

The calculus we are going to present will be generic for most of the classes of Kripke frames from Definition 40. The notion of accessibility between prefixes will play a crucial role. In fact, the rules will be the same in all cases, but will refer to different definitions of the accessibility relation.

Definition 56 (Prefix Accessability) *Let* σ*,* σ' *be prefixes.*

σ' *is* $\mathbf{K}$*-accessible from* σ	*iff*	$\sigma' = \sigma n$ *for some* $n \in \mathbb{N}$
σ' *is* $\mathbf{T}$*-accessible from* σ	*iff*	$\sigma' = \sigma$ *or* $\sigma' = \sigma n$ *for some* $n \in \mathbb{N}$
σ' *is* $\mathbf{S4}$*–accessible from* σ	*iff*	σ *is an initial segment of* σ'
σ' *is* $\mathbf{K4}$*–accessible from* σ	*iff*	σ *is a strict initial segment of* σ' *i.e.,*$\sigma \neq \sigma'$

To reduce the number of case distinction in definitions and proofs we use the usual classification of signed formulas. In the case classical propositional logic there are α, β formulas, which we intuitively think of as *conjunctive* and *disjunctive* formulas. Both types are made up of two constituent formulas α_0, α_1 and β_0, β_1. For modal logic we need the additional categories of ν and π formulas which we may intuitively think of as *Box* formulas and *Diamond* formulas. Both of these types of one constituent formula ν_0 respectively π_0.

Definition 57 (Classification of Signed Modal Formulas)

α*-formulas:*		α_0	α_1
	$TA \wedge B$	TA	TB
	$FA \vee B$	FA	FB
	$FA \to B$	TA	FB
	$F\neg A$	TA	TA
β*-formulas:*		β_0	β_1
	$TA \vee B$	TA	TB
	$FA \wedge B$	FA	FB
	$TA \to B$	FA	TB
	$T\neg A$	FA	FA
ν*-formulas:*		ν_0	
	$T\Box A$	TA	
	$F\Diamond A$	FA	
π*-formulas:*		π_0	
	$T\Diamond A$	TA	
	$F\Box A$	FA	

This classification is exhaustive in the sense that every signed modal formula is either an α-, β-, ν-, or π- formula or it is a signed atom.

Let σ be an arbitrary prefix, α a signed α-formula, β a signed β-formula, ν a signed ν-formula, π a signed π-formula.

$$\frac{\sigma\alpha}{\begin{array}{c}\sigma\alpha_0\\ \sigma\alpha_1\end{array}} \qquad \frac{\sigma\beta}{\sigma\beta_0 \text{ or } \sigma\beta_1}$$

$$\frac{\sigma\nu}{\sigma'\nu_0}$$
σ' accessible from σ and
σ' occurs already on the branch

$$\frac{\sigma\pi}{\sigma'\pi_0}$$
σ' accessible from σ and
σ' is not an initial segment of a
prefix occuring on the branch

Figure 3.5: Modal Tableau Rules

Definition 58 (Modal Tableau) *A modal tableau is a tree, whose nodes are labeled by signed prefix formulas. Not every tree of this kind is a tableau.*

1. *A single node with arbitrary label is tableau, called an initial tableau.*
2. *If T is a tableau, B a branch in T, A a signed prefix formula, that is not a signed atomic formula, occuring somewhere on the branch B. Then the tableau(x) T_1 (T_2) obtained from T by extending B according to the rules in Figure 3.5 are also tableau(x).*
   Note, that if A is a β-formula there will be two continuations T_1 and T_2. In all other cases there is one continuation T_1.

A branch is called closed *if it contains two formulas of the form σFG and σTG*

A tableau is called closed *if all its branches are closed.*

A tableau is called a X-tableau if the accessibility relation in the ν- and π rules is X-accessibility for one of the accessibility modi from Definition 56. If no X is specified, we assume K-accessibilty.

Let us look at a few examples to gain some familiarity with constructing tableaux.

Example 14

$\langle 1 \rangle$	$F(\Box A \wedge \Box B) \rightarrow \Box(A \wedge B)$	(1)
$\langle 1 \rangle$	$T\Box A \wedge \Box B$	(2) *from* 1
$\langle 1 \rangle$	$F\Box(A \wedge B)$	(3) *from* 1
$\langle 1 \rangle$	$T\Box A$	(4) *from* 2
$\langle 1 \rangle$	$T\Box B$	(5) *from* 2
$\langle 11 \rangle$	$FA \wedge B$	(6) *from* 3
$\langle 11 \rangle$	TA	(7) *from* 4
$\langle 11 \rangle$	TB	(8) *from* 5

$\langle 11 \rangle$ FA (9) *from* (6) *closed* $(9, 7)$

$\langle 11 \rangle FB$ (10) *from* (6) *closed* $(10, 8)$

Example 15

$\langle 1 \rangle$	$F\Box(A \wedge B) \rightarrow (\Box A \wedge \Box B)$	(1)
$\langle 1 \rangle$	$T\Box(A \wedge B)$	(2) *from* 1
$\langle 1 \rangle$	$F\Box A \wedge \Box B$	(3) *from* 1

$\langle 1 \rangle$	$F\Box A$	(4) *from* 3
$\langle 11 \rangle$	FA	(6) *from* 4
$\langle 11 \rangle$	$TA \wedge B$	(7) *from* 2
$\langle 11 \rangle$	TA	(8) *from* 7
$\langle 11 \rangle$	TB	(9) *from* 7
	closed	$(8, 6)$

$\langle 1 \rangle$	$F\Box B$	(5) *from* 3
$\langle 11 \rangle$	FB	(10) *from* 5
$\langle 11 \rangle$	$TA \wedge B$	(11) *from* 2
$\langle 11 \rangle$	TA	(12) *from* 11
$\langle 11 \rangle$	TB	(13) *from* 11
	closed	$(13, 10)$

Note: formula (2) is used twice.

Example 16

$\langle 1 \rangle$	$F\Box(A \vee B) \rightarrow (\Box A \vee \Box B)$	(1)
$\langle 1 \rangle$	$T\Box(A \vee B)$	(2) *from* 1
$\langle 1 \rangle$	$F\Box A \vee \Box B$	(3) *from* 1
$\langle 1 \rangle$	$F\Box A$	(4) *from* 3
$\langle 1 \rangle$	$F\Box B$	(5) *from* 3
$\langle 11 \rangle$	FA	(6) *from* 4
$\langle 12 \rangle$	FB	(7) *from* 5
$\langle 11 \rangle$	$TA \vee B$	(8) *from* 2
$\langle 12 \rangle$	$TA \vee B$	(9) *from* 2

$\langle 11\rangle TA \quad (10)[8]$	$\langle 11\rangle TB \quad (11)[8]$	
	$\langle 12\rangle TA \quad (12)[9]$	$\langle 12\rangle TB \quad (13)[9]$
closed $(10, 6)$	$\vdots$	*closed* $(13, 7)$
	open	

Example 17 *(S4-tableau)*

$\langle 1\rangle$	$F\Box A \to \Box(\Box A \vee B)$	(1)
$\langle 1\rangle$	$T\Box A$	(2) *from* 1
$\langle 1\rangle$	$F\Box(\Box A \vee B)$	(3) *from* 1
$\langle 11\rangle$	$F\Box A \vee B$	(4) *from* 3
$\langle 11\rangle$	$F\Box A$	(5) *from* 4
$\langle 11\rangle$	FB	(6) *from* 4
$\langle 111\rangle$	FA	(7) *from* 5
$\langle 111\rangle$	TA	(8) *from* 2

In the **K***-calculus we could in line 8 only obtain* $\langle 11\rangle\ TA$ *which does not lead to a closed tableau.*

You might have guessed it by now or you may have known it already from previous encounters with the tableau calculus: a formula A is a tautology if there is a closed tableau with root label $\langle 1\rangle FA$. Here the signed formula part FA is important. It does not make any difference with what prefix we start. We will now step by step confirm your guess.

Definition 59 *A tableau* T *is* satisfiable*, if there is a branch* B *in* T*, a Kripke structure* (G, R, v) *and a mapping* I *from the set of all prefixes occuring in* B *into the set* G *of possible worlds such that*

$$R(I(\sigma), I(\sigma')) \quad \text{if } \sigma' \text{ is accessible from } \sigma$$

and

$$I(\sigma) \models A \qquad \text{for every formula } \sigma A \text{ in } B.$$

Theorem 69 (Correctness Theorem for Modal Logic K)
Assume that there exists a closed tableau with root label $1\ FB$*, then* B *is a tautology.*

Proof Let T be the closed tableau with root label $\langle 1\rangle FB$. Assume for the sake of a contradiction that there is a Kripke structure (G, R, v) and $g \in G$ with $g \models \neg B$. The tableau T_0 consisting only of the root node of T would thus be satisfiable according to Definition 59 with $I(1) = g$. By the Correctness Lemma 70 also T had to be satisfiable. But, this contradicts the assumption that T is closed.

∎

Lemma 70 (Correctness Lemma) *If T_0 is a satisfiable tableau and T is obtained from T_0 by a rule application from Figure 3.5 then also T is satisfiable.*

Proof The proof proceeds by case distinction according to which rule is used to obtain T from T_0. Since the correctness of the α and β rules has already been dealt with in classical propositional logic we will only present tha cases for the ν- and π rules.

ν-case
There is thus a branch P in T_0 and a formula $\sigma\nu$ on P such that T is obtained by extending P by a new node labeled with $\sigma'\nu_0$ such that σ' is accessible from σ and σ' already occurs on P. If the satisfiable branch of T_0 is different from P, then T is trivially also satisfiable. So let us assume the P is satisfiable in the sense of Definition 59. There is thus a Kripke structure (G, R, v) and a mapping I satisfying

$$R(I(\sigma), I(\sigma')) \quad \text{as well as} \quad I(\sigma) \models \nu.$$

This implies $I(\sigma') \models \nu_0$. (Think of the typical case $\nu = T\Box A$.)

π-case
By the case assumption there is a formula $\sigma\pi$ on the branch P and T is obtained by extending P by a new node labeled with $\sigma'\pi_0$. Again we may assume that P is a satisfiable branch and thus know that there is a Kripke structure (G, R, V) and a mapping I such that $I(\sigma) \models \pi$. Thus implies (think again of the typical case $\pi = T\Diamond\pi_0$) the existence of a world $g \in G$ with $R(I(\sigma), g)$ and $g \models \pi_0$. Since σ' is new on P it is not in the domain of I. We may thus extend I to I' by defining $I(\sigma') = g$. We want to argue

that this definition of I' shows that branch P in T is satisfiable. Checking Definition 59 we notice that it remains to prove:

1. for any ρ occuring on P :
   if σ' is accessible from ρ then $R(I'(\rho), I'(\sigma'))$

2. for any ρ occuring on P :
   if ρ is accessible from σ' then $R(I'(\sigma'), I'(\rho))$

We observe that case 1 is only possible for $\rho = \sigma$ and we know $R(I(\sigma), I(\sigma'))$. In case 2 σ' would be an initial segment of ρ which is forbidden by the side condition of π-rule application.

∎

Example 18 *(**K**-tableau)*

$$\begin{array}{lll} \langle 1\rangle & F\Box A \to \Diamond A & (1) \\ \langle 1\rangle & T\Box A & (2)\ \textit{from}\ 1 \\ \langle 1\rangle & F\Diamond A & (3)\ \textit{from}\ 1 \end{array}$$

Formulas (2) and (3) are both ν-formulas. No further rule is applicable.

Let us drop for a moment the restriction on the ν-rule that only prefixes σ' already present on the branch are allowed. Then we could continue the tableau constructions

$$\begin{array}{lll} \langle 1\rangle & F\Box A \to \Diamond A & (1) \\ \langle 1\rangle & T\Box A & (2)\ \textit{from}\ 1 \\ \langle 1\rangle & F\Diamond A & (3)\ \textit{from}\ 1 \\ \langle 11\rangle & TA & (4)\ \textit{from}\ 2 \\ \langle 11\rangle & FA & (5)\ \textit{from}\ 3 \end{array}$$

and arrive at a closed tableau. The formula $\Box A \to \Diamond A$ is not a K-tautology. We have thus seen that the restriction on the ν-rule is neccessary to obtain a correct calculus.

Theorem 71 (Completeness Theorem for Modal Logic K)
If A is a tautology then there is a closed tableau with root label $\langle\alpha\rangle FA$ for an arbitrary prefix α.

Proof We assume that the reader has already seen a completeness proof for a tableau calculus. So we will be a bit sketchy about the general structure of the proof and concentrate on the specifics of this calculus.

The general pattern is as follows. Assume there is no closed tableau for the initial tableau with label $\langle\alpha\rangle FA$. We will construct a Kripke structure (G, R, v) and a world $g \in G$ such that $g \models \neg A$. This contradict the assumption that A is a tautology.

Now, for the construction itself. Assume we have exhausted all possibilities of finding a closed tableau with root label $\langle\alpha\rangle FA$. Since we need to take into account that the ν-rule may be applied an unknown number of times we arrive at an infinite tableau T_∞ with at least one open branch P. Let G be the set of prefixes occuring on P and the accessibility relation R defined for $\sigma, \sigma' \in G$ by

$$R(\sigma, \sigma') \Leftrightarrow \sigma' \text{ is accessible from } \sigma \text{ as a prefix}$$

For $\sigma \in G$ and p a propositional atom we define

$$v(\sigma, p) = 1 \Leftrightarrow \text{ if } \sigma Tp \text{ occurs in } P$$

This completes the definition of the Kripke structure (G, R, v). We will next show that for all $\sigma \in G$

$$\text{if } \sigma XB \text{ is in } P \text{ then } ((G, R, v), \sigma) \models XB \text{ for } X \in \{T, F\} \tag{3.10}$$

Since the root is in this case in P this will in particular entail $((G, R, v), \alpha) \models FA$, i.e., $((G, R, v), \alpha) \models \neg A$ and we will be finished.

The proof of 3.10 proceeds by structural induction on B.

The initial cases are σTp and σFp. If σTp occurs in P we have by definition $((G, R, v), \sigma) \models p$. If σFp occurs in P we know that σTp does not occur in P, since P was assumed to be an open branch. This yields again by the same definiton $((G, R, v), \sigma) \models \neg p$.

We skip the cases where the leading connective of B is a propositional connective.

$\mathbf{B = \Box B_0}$ Assume $\sigma \Box B_0$ occurs in P. For any $\sigma' \in G$ that is accessible from σ the ν rule for $\sigma \Box B_0$ becomes at some point applicable and puts $\sigma' B_0$ into P. By induction hypothesis this gives us $((G, R, v), \sigma') \models B_0$ and since

this works for any σ' of the given kind we also obtain $((G, R, v), \sigma) \models \Box B_0$. As desired.

$\mathbf{B = \Diamond B_0}$ If $\sigma \Diamond B_0$ occurs in P an application of the π-rule guarantees a σ' that is accessible from σ such that $\sigma' B_0$ occurs in P. By induction hypothesis $((G, R, v), \sigma') \models B_0$ and this also $((G, R, v), \sigma) \models \Diamond B_0$

▪

So far the tableau calculus can only be used to prove tautologies. We are going to present extensions for dealing with local and global logical inference, see Definition 42.

Definition 60 *Let $M \subseteq PModFml$ be a set of modal formulas.*

1. *A local tableau for M is a tableau T built up from an initial tableau $\sigma_0 X A$ by using in addition to the rules from Figure 3.5 also the following rule:*

$$\frac{}{\sigma_0 T A} \qquad \text{for every } A \in M$$

2. *A global tableau for M is a tableau T built up from an initial tableau by using in addition to the rules from Figure 3.5 also the following rule:*

$$\frac{}{\sigma T A} \qquad \text{for every } A \in M \text{ and any prefix } \sigma \text{ occuring on the path}$$

Corollary 72 *Let $M \subseteq PModFml$ be a set of modal formulas.*

1. *$M \vdash_L A$ iff there is a closed local tableau for M with root label $\sigma_0 F A$.*
2. *$M \vdash_G A$ iff there is a closed global tableau for M with root label $\sigma_0 F A$.*

Proof to be done

We still need to address an important phenomenon that does not arise for tableau calculi for classical propositional logic, but does here. To pinpoint

the problem look at the following tableau construction in the $S4$-calculus.

$\langle 1 \rangle$	$F \Box \Diamond p \rightarrow \Diamond \Box p$	(1)
$\langle 1 \rangle$	$T \Box \Diamond p$	(2)[1]
$\langle 1 \rangle$	$F \Diamond \Box p$	(3)[1]
$\langle 1 \rangle$	$T \Diamond p$	(4)[2]
$\langle 1 \rangle$	$F \Box p$	(5)[3]
$\langle 11 \rangle$	$T p$	(6)[4]
$\langle 12 \rangle$	$F p$	(7)[5]
$\langle 11 \rangle$	$T \Diamond p$	(8)[2]
$\langle 11 \rangle$	$F \Box p$	(9)[3]
$\langle 12 \rangle$	$T \Diamond p$	(10)[2]
$\langle 12 \rangle$	$F \Box p$	(11)[3]
$\langle 111 \rangle$	$T p$	(12)[8]
$\langle 112 \rangle$	$F p$	(13)[9]
$\langle 121 \rangle$	$T p$	(14)[10]
$\langle 122 \rangle$	$F p$	(15)[11]

It is evident that this tableau construction does not terminate. And since $\Box \Diamond p \rightarrow \Diamond \Box p$ is not a S4-tautology the correctness theorem says that there cannot be a closed tableau. But, in contrast to tableau calculi for classical propostional logic the search for a closed tableau does not terminate.

Lemma 73 *Let T be a tableau with root label $\sigma_0 X A$.*
Then for every signed prefix formula $\sigma Y B$ occuring as a label in T we know B is a subformula of A.

Proof This can be easily verified by inspecting the rules in Figure 3.5.

3.6 Description Logic

The origins of description logics can be traced back to the systems for knowledge representation within artificial intelligence in the 1970s. From the first rather ad-hoc and not very formal beginnings knowledge representation established its own community but more and more using concepts from traditional logic. A comprehensive survey of the state of the art can be found in [Baader *et al.*, 2003].

There is a great number of variants of description logic, see e.g., [Baader *et al.*, 2003, Chapters 3 and 5]. We will first present a simple, prototypical example called $\mathcal{ALC}$ and later a more expressive variant called $\mathcal{SHIQ}$. There is a kind of systematic naming scheme for description logics. $\mathcal{AL}$ stands for *attributive language* and appending $\mathcal{C}$ signifies that negation (complement) is allowed. Other possible extensions could be $\mathcal{ALCN}$ which would allow *number restrictions* or $\mathcal{ALCE}$ which would allow *unrestricted extistential quantification*.

Basic Description Logic

Description logic is based in the two fundamental notions of *concept* and *role*. The are many variations on what kind of roles may be used and on the ways expressions can be composed. We will first present the simplest description logic $\mathcal{ALC}$. The syntactical material from which $\mathcal{ALC}$ is built is determined by a set $\mathbf{C}$ of concept symbols and a set $\mathbf{R}$ of symbols for roles. The union $\mathbf{V} = \mathbf{C} \cup \mathbf{R}$ makes up the *vocabulary* for a specific instance of $\mathcal{ALC}$. Of course expressions will also contain built-in or logical symbols as explained in the following definition.

Definition 61 ($\mathcal{ALC}$-Expressions)
The set of $\mathcal{ALC}$ expressions for a given vocabulary $\mathbf{V} = \mathbf{C} \cup \mathbf{R}$ is inductively defined by:

1. *every concept symbol C from $\mathbf{C}$ is an $\mathcal{ALC}$ expression,*
2. *if $C_1, \ldots, C_k$ are $\mathcal{ALC}$ expressions so are $C_1 \sqcap \ldots \sqcap C_k$, $C_1 \sqcup \ldots \sqcup C_k$ and $\neg C_1$,*

3. *if C is a $\mathcal{ALC}$ expression, R a role symbol from $\mathbf{R}$, then $\exists R.C$ and $\forall R.C$ are also $\mathcal{ALC}$ expressions.*

Sometimes the expressions definied in Definition 61 are called *concept expressions*. Since in the simple description logic $\mathcal{ALC}$ there are no other expressions there is no need to be specific.

For later reference we define some syntactic properties and operations on concept expressions.

Definition 62
Let C be a concept expression. The role depth $rd(C)$ of C is the maximal number of nestings of role operators in C. Formally:

$$\begin{array}{lcll} rd(C) & = & 0 & \text{if } C \in \mathbf{C} \\ rd(\neg C) & = & rd(C) & \\ rd(C_1 \sqcup C_2) & = & max(rd(C_1), rd(C_2)) & \\ rd(C_1 \sqcap C_2) & = & max(rd(C_1), rd(C_2)) & \\ rd(\forall R.C) & = & rd(C) + 1 & \\ rd(\exists R.C) & = & rd(C) + 1 & \end{array}$$

Definition 63
The set of all subexpressions $SubEx(C)$ of an expression C is inductively defined by

$$\begin{array}{lcll} SubEx(C) & = & \{C\} & \text{if } C \in \mathbf{C} \\ SubEx(\neg C) & = & SubEx(C) \cup \{\neg C\} & \\ SubEx(C_1 \sqcup C_2) & = & SubEx(C_1) \cup SubEx(C_2) \cup \{C_1 \sqcup C_2\} & \\ SubEx(C_1 \sqcap C_2) & = & SubEx(C_1) \cup SubEx(C_2) \cup \{C_1 \sqcap C_2\} & \\ SubEx(\forall R.C) & = & SubEx(C) \cup \{\forall R.C\} & \\ SubEx(\exists R.C) & = & SubEx(C) \cup \{\exists R.C\} & \end{array}$$

We call a concept expression of the form $\exists R.C$ an existential expression.

What do concept expressions stand for? In particular it is not clear what the expressions involving quantifiers stand for. Before we answer this question we need to fix the *semantic domain*, i.e., we fix the domain that we will use to define the meaning of concept expressions. Roughly speaking, concept symbols will denote sets of objects and roles will denote binary relations on objects. More formally

Definition 64 (Interpretation)
We first fix a domain of objects, usually denoted by Δ. That is the universe of discourse and corresponds to the universe of structures for first-order logic. An interpretation for $\mathcal{ALC}$ for the vocabulary $\mathbf{V}$ is a mapping $\mathcal{I}$ such that

1. *for every concept symbol $C \in \mathbf{C}$ the interpretation $C^{\mathcal{I}}$ is a subset of Δ*
2. *for every role symbol $R \in \mathbf{R}$ the interpretation $R^{\mathcal{I}}$ is a binary relatin of Δ, i.e. $R^{\mathcal{I}} \subseteq \Delta \times \Delta$.*

Now we are well equipped to define the semantics of $\mathcal{ALC}$ expressions.

Definition 65 (Semantics of $\mathcal{ALC}$)
Let $\mathcal{I}$ be an interpretation.

1. $(C_1 \sqcap \ldots \sqcap C_k)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap \ldots \cap C_k^{\mathcal{I}}$
2. $(C_1 \sqcup \ldots \sqcup C_k)^{\mathcal{I}} = C_1^{\mathcal{I}} \cup \ldots \cup C_k^{\mathcal{I}}$
3. $(\neg C)^{\mathcal{I}} = \Delta \setminus C^{\mathcal{I}}$
4. $(\exists R.C)^{\mathcal{I}} = \{d \in \Delta \mid$ *there exists* $(d, e) \in R^{\mathcal{I}}$ *with* $e \in C^{\mathcal{I}}\}$,
5. $(\forall R.C)^{\mathcal{I}} = \{d \in \Delta \mid$ *for all* $(d, e) \in R^{\mathcal{I}}$ *it is true that* $e \in C^{\mathcal{I}}\}$

Example 19
Let M, W, P be symbols for concepts in $\mathbf{C}$ that we intuitively think of as the class of all men, all women and all persons. Furthermore, there is a role symbol R in $\mathbf{R}$. We think of $R(a, b)$ as a is a direct ancestor of b. Then the following are correct expressions in the vocabulary $\mathbf{V} = \mathbf{C} \cup \mathbf{R}$ defining the derived concepts of father Fa, mother Mo and parent Pa.

$$Fa = M \sqcap \exists R.P \qquad Mo = W \sqcap \exists R.P \qquad Pa = Fa \sqcup Mo$$

Let us compare the concept definitions from Example 19 with the corresponding definitions in first-order predicate logic:

$$Fa(x) = M(x) \wedge \exists y(R(x, y) \wedge P(y))$$
$$Mo(x) = W(x) \wedge \exists y(R(x, y) \wedge P(y))$$
$$Pa(x) = Fa(x) \vee Mo(x)$$

The most striking difference is that in first-order logic the free and quantified variables, x and y in this case, occur explicitely, in description logic they are present only implicitely.

So far we have introduced $\mathcal{ALC}$ expressions that denote sets of objects. We will now introduce a way to state claims involving these concept expressions.

Definition 66 (Concept Formulas)
For any two $\mathcal{ALC}$ expressions C_1, C_2 the following will be $\mathcal{ALC}$ formulas:

$C_1 = C_2$	*equality problem*
$C_1 \sqsubseteq C_2$	*subsumption problem*
$C_1 = \emptyset$	*emptyness problem*

As the last item on the checklist for introducing a new logic we define validity. There is no surprise here.

Definition 67 *Let Γ be a set of concept formulas, F a single concept formula and $\mathcal{I}$ an interpretation.*

1. *We say $\mathcal{I}$ satisfies F, written as $\mathcal{I} \models F$, if*

   $\mathcal{I} \models C_1 = C_2$ *iff* $C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$
   $\mathcal{I} \models C_1 \sqsubseteq C_2$ *iff* $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
   $\mathcal{I} \models C_1 = \emptyset$ *iff* $C_1^{\mathcal{I}} = \emptyset$

2. Γ *is* valid*, if for all interpretations $\mathcal{I}$ we have $\mathcal{I} \models F$ for all $G \in \Gamma$.*

3. *C_1 and C_2 are called equivalent if $C_1 = C_2$ is valid.*

4. Γ *is* satisfiable*, if there is an interpretation $\mathcal{I}$ such that $\mathcal{I} \models F$ for all $G \in \Gamma$.*

It can be easily seen that for any interpretation $\mathcal{I}$ we have:

$$\mathcal{I} \models C_1 = C_2 \text{ iff } \mathcal{I} \models C_1 \sqsubseteq C_2 \wedge C_2 \sqsubseteq C_1$$

and

$$\mathcal{I} \models C_1 \sqsubseteq C_2 \text{ iff } \mathcal{I} \models C_1 \sqcap \neg C_2 = \emptyset$$

Thus it suffices to solve the emptyness problem. The other two problem types can be reduced to it.

It is a suprising fact that description logic can be reduced to multi-modal logic. This was first observed in [Schild, 1991, Schild, 1993]. We present how this reduction works.

Definition 68 (Translation into Modal Logic)
Every $\mathcal{ALC}$ expression F in the vocabulary $\mathbf{V} = \mathbf{C} \cup \mathbf{R}$ will be translated into a multi-modal formula F^* with $\mathbf{C}$ as propositional atoms and modal operators $\Box_R, \Diamond_R$ for every $R \in \mathbf{R}$ as follows:

$$\begin{array}{lcll} F^* & = & F & \text{if } F \text{ is a concept symbol in } \mathbf{C} \\ (F_1 \sqcap F_2)^* & = & (F_1^* \wedge F_2^*) & \\ (\neg F)^* & = & \neg F^* & \\ (\forall R.F)^* & = & \Box_R F^* & \\ (\exists R.F)^* & = & \Diamond_R F^* & \end{array}$$

We do have the feeling that F^* conveys in a different way the same information as F. We will take the time to make this precise.

The first step is to relate the semantics of description logic with the semantics of multi-modal logic, i.e., to find a correspondence between interpretations $\mathcal{I}$ and Kripke structures $\mathcal{K}$.

Definition 69 (Relating Intepretations and Kripke structure)

1. *For any interpretation $\mathcal{I}$ with domain of objects Δ we associate the Kripke structure $\mathcal{K}_\mathcal{I} = (G_\mathcal{I}, \{R_\mathcal{I} \mid R \in \mathbf{R}\}, v_\mathcal{I})$:*

$$\begin{array}{lcl} G_\mathcal{I} & = & \Delta \\ R_\mathcal{I} & = & R^\mathcal{I} \\ v_\mathcal{I}(d, C) = 1 & \Leftrightarrow & d \in C^\mathcal{I} \end{array}$$

2. *For every multi-modal Kripke structure $\mathcal{K} = (G, \{R \mid R \in \mathbf{R}\}, v)$ we associate an interpretation $\mathcal{I}_\mathcal{K}$:*

$$\begin{array}{lcl} \Delta & = & G \\ R^{\mathcal{I}_\mathcal{K}} & = & R \\ C^{\mathcal{I}_\mathcal{K}} & = & \{d \in G \mid v(d, C) = 1\} \end{array}$$

Now, the next lemma precisely states the relation between F and F^*.

Lemma 74 (Translation Lemma)

1. *For every $\mathcal{ALC}$ expression F and every interpretation $\mathcal{I}$*

$$F^{\mathcal{I}} = \{d \in \Delta \mid (\mathcal{K}_{\mathcal{I}}, d) \models F^*\}$$

2. *There is an interpretation $\mathcal{I}$ and an object $d \in \Delta$ satisfying $d \in F^{\mathcal{I}}$ iff F^* is satisfiable.*

3. *$F_1 \sqsubseteq F_2$ is valid if and only if $(F_1 \sqcap \neg F_2)^*$ is not satisfiable.*

Proof
ad 1: As was to be expected the proof proceeds by structural induction on F. If F is a concpet name we get by definition of $\mathcal{K}_{\mathcal{I}}$

$$\begin{array}{rcl} \{d \in \Delta \mid (\mathcal{K}_{\mathcal{I}}, d) \models F^*\} & = & \{d \in \Delta \mid v(d, F) = 1\} \\ & = & F^{\mathcal{I}} \end{array}$$

The induction steps for the operators $\sqcap$,$\sqcup$ und $\neg$ are simple and will be omitted here. We take on $F = \forall R.F_1$, the case $F = \exists R.F_1$ is completely analogous.

$$\begin{array}{rcl} \{d \in \Delta \mid (\mathcal{K}_{\mathcal{I}}, d) \models (\forall R.F_1)^*\} & = & \{d \in \Delta \mid (\mathcal{K}_{\mathcal{I}}, d) \models \Box_R F_1^*\} \\ & = & \{d \in \Delta \mid \text{ for all } e \text{ with } R(d, e) \\ & & \quad (\mathcal{K}_{\mathcal{I}}, e) \models F_1^* \text{ is true } \} \\ & = & \{d \in \Delta \mid \text{ for all } e \text{ with } R(d, e) \\ & & \quad e \in F_1^{\mathcal{I}} \text{ is true } \} \\ & = & (\forall R.F_1)^{\mathcal{I}} \end{array}$$

ad 2: Is an immediate consequence of part 1 and Definition 69.

ad 3: $F_1 \sqsubseteq F_2$ is valid if and only if, for all $\mathcal{I}$ and all $d \in \Delta$ we get $d \notin (F_1 \sqcap \neg F_2)^{\mathcal{I}}$. Using part 1 of this lemma this is equivalent to $(K^{\mathcal{I}}, d) \not\models F_1^* \wedge \neg F_2^*$ for all $\mathcal{I}$ and all $d \in \Delta$. But, this is to say that $(F_1 \sqcap \neg F_2)^*$ is not satisfiable.

∎

We have so far gained a deeper understanding of the notions of concept and role that are so essential to knowledge representation systems in general and description logics in particular. It is now time to address another fundamental way, that originated within the artificial intelligence movement, to structure the representation of knowledge. It distinguishes between *terminological knowledge*, that is collected in what is called the T-box and *assertional knowledge* that is collected in the A-box.

Definition 70 (T-Box)
The set $\mathbf{C}$ *of concept symbols is divided into* $\mathbf{C} = \mathbf{C}_b \cup \mathbf{C}_n$ *with* $\mathbf{C}_b$ *the set of* base concept symbols *and* $\mathbf{C}_n$ *the set of* name symbols.
A T-Box $\mathcal{T}$ *is a set of equations* $C_1 = C_2$ *where* $C_1 \in \mathbf{C}_n$ *and* C_2 *a concept expression using only concept from* $\mathbf{C}$. *For every* $C_1 \in \mathbf{C}_n$ *there is at most one equation in* $\mathcal{T}$ *with lefthand side* C_1.
A concept $C_1 \in \mathbf{C}_n$ directly uses $B \in \mathbf{C}$ *in* $\mathcal{T}$ *if* $C_1 = C_2$ *is in* $\mathcal{T}$ *and* B *occurs in* C_2.
A concept $C_1 \in \mathbf{C}_n$ uses B *if* (C_1, B) *is in the transitive closure of the direct use relation.*
A T-Box $\mathcal{T}$ *is called* cyclic *if there is* $C \in \mathbf{C}_n$ *that uses itself and* acyclic *if no such* C *exists.*

Sometimes base concepts are also called *primitive concepts* and name concepts are refered to as *defined concepts*.

One would expect that an interpretation $\mathcal{I}$ satisfies a T-Box $\mathcal{T}$ if $C^{\mathcal{I}} = C_1^{\mathcal{I}}$ for every equation $C = C_1$ in $\mathcal{T}$. This is at least one option. Within the knowledge representation community it is customary to view a T-Box as a set of definitions and these definitions should be unique that is to say, once the interpretation of the base concepts and roles are fixed there is only one way to satisfy the definitions in the T-Box.

Definition 71 (Definitorial T-Box)
A T-Box $\mathcal{T}$ *is called* definitorial *if for any two interpretations* $\mathcal{I}$ *and* $\mathcal{J}$ *over the same domain* Δ *satisfying* $\mathcal{T}$ *and with* $C^I = C^J$ *for all base concepts* $C \in \mathbf{C}_b$ *and* $R^I = R^J$ *for all roles* $R \in \mathbf{R}$ *we already have* $\mathcal{I} = \mathcal{J}$, *that is* $C^I = C^J$ *for all name concepts* $C \in \mathbf{C}_n$.

It is easy to see that any acyclic T-Box is definitorial, see Exercise 3.9.19. The problem arises with *recursive* definitions $C = C_1$ of a name concept C

where the defined concept also occurs in C_1. The standard view is that such an equation defines C as the least fixpoint. It can be shown that in the present context such fixpoints always exist. For the details that are a little bit more involved since in general we have to consider a system of mutual recursive equations see [Baader *et al.*, 2003, pp 59ff]. We will restrict our attention to acyclic terminologies.

Definition 72 (A-Box)
Let K be a set of constant symbols.
An A-Box $\mathcal{A}$ is a set of formulas of the form

$$C(c), \quad R(c,d)$$

with C a concept expression, $R \in \mathbf{R}$ and c, d constants in K.

This definition introduces the new concept of a constant symbol. This requires an appropriate extension of the definition of an interpretation $\mathcal{I}$ (Definition 64). It is intuitively clear that we want $\mathcal{I}(c) \in \Delta$ for any $c \in K$. In the context of knowledge representation systems it is common practice to assume the *unique naming assumption*. This requires that $\mathcal{I}(c) \neq \mathcal{I}(d)$ if c and d are different symbols in K.

Notice, that while the arguments of concept symbols and roles are suppressed in the T-box they are explicitly present in the A-box.

A Tableau Calculus for $\mathcal{ALC}$

We assume that the reader has already acquired some familiarity with tableau calculi.

Before we go into the details of constructing tableau proofs let us first take stock of what questions we want to be answered.

Definition 73
Let $\mathcal{T}$ be an acyclic T-box, $\mathcal{A}$ an A-box, and C' an $\mathcal{ALC}$-expression.

1. *C' is satisfiable with respect to $\mathcal{T}$,*
   i.e., if there is an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} = C_1^{\mathcal{I}}$ for every equation $C = C_1$ in $\mathcal{T}$ and $(C')^{\mathcal{I}} \neq \emptyset$.

2. *$\mathcal{A}$ is consistent with respect to $\mathcal{T}$,*
   i.e., there is an interpretation $\mathcal{I}$ such that

   (a) *$C^{\mathcal{I}} = C_1^{\mathcal{I}}$ for every $C = C_1$ in $\mathcal{T}$,*

   (b) *$d^{\mathcal{I}} \in D^{\mathcal{I}}$ for every expression $D(d) \in \mathcal{A}$ and*

   (c) *$(d_1^{\mathcal{I}}, d_2^{\mathcal{I}}) \in R^{\mathcal{I}}$ for every expression $R(d_1, d_2)$ in $\mathcal{A}$*

We will present a tableau calculus that solves the satisfiability problem for $\mathcal{ALC}$-expressions and the consistency problem for A-boxes with respect to acyclic T-boxes.

Our first result reduces the two questions concidered so far to the corresponding questions with respect to an empty T-box. This is similar in a way to the deduction lemmata in classical logic.

Lemma 75 (Eliminating the T-box)
Let $\mathcal{T}$ be an acyclic T-box.

1. *For any concept expression D there is an expression D' such that D is satisfiable with respect to $\mathcal{T}$ iff D' is satisfiable with respect to the empty T-box.*

2. *For any A-box $\mathcal{A}$ there is an A-box $\mathcal{A}'$ such that $\mathcal{A}$ is consistent with respect to $\mathcal{T}$ iff $\mathcal{A}'$ is consistent with respect to the empty T-box.*

Proof
ad 1 We may assume, see Exercise 3.9.19, that all equations in $\mathcal{T}$ are of the form $C = C_1$ where C_1 only contains base symbols. Let D' be obtained from D by replacing each name symbol by its definition. Obviously, D is satisfiable with respect to $\mathcal{T}$ iff D' is satisfiable with respect to $\mathcal{T}$. But, since D' does not contain any name symbols its satisfiability is completeley independent of any T-box.

ad 2 We use the same trick as in part 1. Replace every name symbol D in $\mathcal{A}$ by its definition to obtain the modified A-box $\mathcal{A}'$. Again, $\mathcal{A}$ is consistent with respect to $\mathcal{T}$ iff $\mathcal{A}'$ is consistent with respect to $\mathcal{T}$. But, since $\mathcal{A}'$ does not contain any name symbol its consistency is completely independent of any T-box.

$$\frac{(C_1 \sqcap C_2)(c)}{\begin{array}{c} C_1(c) \\ C_2(c) \end{array}} \qquad \frac{(C_1 \sqcup C_2)(c)}{C_1(c) \quad C_2(c)}$$

$$\frac{(\exists R.C)(c)}{\begin{array}{c} R(c,d) \\ C(d) \end{array}} \quad d \text{ a new constant} \qquad \frac{(\forall R.C)(c)}{\begin{array}{c} R(c,d) \\ \hline C(d) \end{array}}$$

Figure 3.6: Tableau Rules for $\mathcal{ALC}$

▪

A concept expression C is called in negation normal form if the negation symbol $\neg$ occurs in C only in front of concept symbols. From the results of classical propositional logic and Exercise 3.9.16 it follows that for every concept expression C there is an equivalent concept expression C_{nnf} in negation normal form.

Definition 74 ($\mathcal{ALC}$-Tableau)
An $\mathcal{ALC}$-tableau is a tree, whose nodes are labeled by formulas of the form

$$C(c), \quad R(c,d)$$

with C a concept expression in negation normal form, $R \in \mathbf{R}$ and constants $c, d \in K$. For easy reference we will call formulas of this type A-box formulas.
For every internal node N the labels of the successor node(s) of N are determined by the rules from Figure 3.6.
A branch B in an $\mathcal{ALC}$-tableau is closed if B contains nodes with labels $C(a)$ and $\neg C(a)$ for a constant $a \in K$ and a concept symbol $C \in \mathbf{C}$.
An $\mathcal{ALC}$-tableau T is closed if all its branches are closed.

Example 20
It is intuitively clear that the formula $\exists R.(C_1 \sqcap C_2) \sqsubseteq (\exists R.C_1) \sqcap (\exists R.C_2)$ is valid. This is equivalent to the formula $\exists R.(C_1 \sqcap C_2) \sqcap \neg((\exists R.C_1) \sqcap (\exists R.C_2))$ being not satisfiable. The negation normal form of this is $\exists R.(C_1 \sqcap C_2) \sqcap ((\forall R.\neg C_1) \sqcup (\forall R.\neg C_2))$. Thus $(\exists R.C_1 \sqcap C_2) \sqcap ((\forall R.\neg C_1) \sqcup (\forall R.\neg C_2))(c)$ is formula number 0 on the tableau, which continues as follows

	$(\exists R.C_1 \sqcap C_2)(c)$		$1[0]$
	$(\forall R.\neg C_1) \sqcup (\forall R.\neg C_2))(c)$		$2[0]$
	$R(c,d)$		$3[1]$
	$(C_1 \sqcap C_2)(d)$		$4[1]$
	$C_1(d)$		$5[4]$
	$C_2(d)$		$6[4]$
$(\forall R.\neg C_1)(c)$	$7[2]$	$(\forall R.\neg C_2)(c)$	$8[2]$
$\neg C_1(d)$	$9[7,3]$	$\neg C_2(d)$	$10[8,3]$
closed	$[9,5]$	*closed*	$[10,6]$

Theorem 76 (Tableau Soundness)
Let C be a concept expression in negation normal form and T be a closed tableau with root label $C(a)$ then C is not satisfiable.

Proof We need the auxiliary notion of a satisfiable tableau: An $\mathcal{ALC}$-tableau T is called satisfiable if there is a branch B and an interpretation $\mathcal{I}$ that satisfies all the A-box formulas in B.
As mentioned in the paragraph following Definition 72 an interpretation $\mathcal{I}$ has to follow the unique naming convention, i.e., $c^{\mathcal{I}} \neq d^{\mathcal{I}}$ for different constants $c, d \in K$. For new constants, and only for the new constants, introduced during the construction of a tableau we drop this requirement. Different new constant symbols d_1, d_2 may be interpreted by the same element in Δ, $d_1^{\mathcal{I}} = d_2^{\mathcal{I}}$ and they may also have the same interpretation as constants from K, $d_1^{\mathcal{I}} = d^{\mathcal{I}}$.
Following the usual strategy of soundness proofs for tableau calculi we will show that:

> if a tableau T is satisfiable and T_1 is obtained from T by application of a rule from Figure 3.6 then T_1 is also satisfiable.

We will only consider the case that T_1 is obtained from T by application of the $\exists R$ rule. There is a branch B of T that is satisfiable by an interpretation $\mathcal{I}$. Without loss of generality we may assume that the rule is applied to a formula $\exists R.C(d)$ on B and d' the newly introduced constant. Since $d^{\mathcal{I}} \in (\exists R.C)^{\mathcal{I}}$ there is an element $a \in \Delta$ with $R^{\mathcal{I}}(d^{\mathcal{I}}, a)$ and $a \in C^{\mathcal{I}}$. If we extend the interpretation $\mathcal{I}$ to $\mathcal{I}'$ by $\mathcal{I}'(d') = a$ then $\mathcal{I}'$ satisfies the extended branch B_1.

The rest of the argument is as usual. A closed tableau is clearly not satisfiable. Applying the above fact repeatedly we know that the initial tableau consisting only of the root labeled by C is not satisfiable. This is the same thing as saying that C is not satisfiable.

∎

Theorem 77 (Tableau Completeness)
Let C be an unsatisfiable concept expression in negation normal form then there exists a closed tableau with root label C.

Proof Assume by way of contradiction that the tableau with root label C making all efforts cannot be closed. There is thus a tableau T and an open branch B of T such that for all formulas on B the corresponding rule application has been performed. We will later see that T can be chosen to be finite, but at the moment we also have to consider the case that B is infinite. From B we will construct an interpretation $\mathcal{I}$ satisfying C. This contradicts the assumptions of the theorem and will show that a closed tableau with root label C exists.
Now for the construction of $\mathcal{I}$. Let Δ be the set of all constants, new and old, occurring in formulas on B. We set $d^{\mathcal{I}} = d$. Furthermore we define for any $R \in \mathbf{R}$ and $C \in \mathbf{C}$

$R^{\mathcal{I}}(c,d)$ iff $R(c,d)$ occurs in B
$c \in C^{\mathcal{I}}$ iff $C(c)$ occurs in B

By structural induction it is easily proved that for any A-box formula $C(d)$ on B this definition yields $C^{\mathcal{I}} = true$. To get a taste of the proof let us look at the case that $C(d) = \forall R.C_1(d)$ is on B. In order to arrive at $C(d)^{\mathcal{I}} = true$ we consider an element $c \in \Delta$ satisfying $R^{\mathcal{I}}(d,c)$. By definition of $\mathcal{I}$ this can only be the case if $R(d,c)$ occurs on the branch B. Now the $\forall R$ rule is applicable and since we assumed that all rule applications have been exhausted we know that $C_1(c)$ is in B. By induction hypothesis $C_1(c)^{\mathcal{I}} = true$ which finishes the argument.

∎

Example 21
We look here at the subsumption relation $\exists R.C_1 \sqcap \exists R.C_2 \sqsubseteq \exists R.(C_1 \sqcap C_2)$ inverse to that considered in Exercise 20. It is equivalent to $\exists R.C_1 \sqcap \exists R.C_2 \sqcap$

$\neg\exists R.(C_1 \sqcap C_2) = \emptyset$ *or in negation normal form* $\exists R.C_1 \sqcap \exists R.C_2 \sqcap \forall R.(\neg C_1 \sqcup \neg C_2) = \emptyset$.

		$\exists R.C_1 \sqcap \exists R.C_2 \sqcap \forall R.(\neg C_1 \sqcup \neg C_2)(c)$			$0[]$
		$\exists R.C_1(c)$			$1[0]$
		$\exists R.C_2(c)$			$2[0]$
		$\forall R.(\neg C_1 \sqcup \neg C_2)(c)$			$3[0]$
		$R(c, d_1)$			$4[1]$
		$C_1(d_1)$			$5[1]$
		$R(c, d_2)$			$6[2]$
		$C_2(d_2)$			$7[2]$
		$(\neg C_1 \sqcup \neg C_2)(d_1)$			$8[3, 4]$
		$(\neg C_1 \sqcup \neg C_2)(d_2)$			$9[3, 6]$
$\neg C_1(d_1)$	$10[8]$		$(\neg C_2)(d_1)$		$11[8]$
closed	$[10, 5]$	$\neg C_1(d_2)$	$12[9]$	$\neg C_2(d_2)$	$13[9]$
		open		*closed*	$[13, 7]$

According to our understanding of tableau rules we could in the previous example again apply the $\exists R$ rule on lines 1 and 2 and obtain new elements d_2 and d_3 and so on. The correctness theorem tells us that we will never reach a closed tableau. We will proof that there is a bound n , that can be easily computed from the root label, such that if after n steps we have not reached a closed tableau we need not look any further.

The first step towards this goal is a more reglemented application of tableau rules.

Definition 75 (Optimized $\mathcal{ALC}$-Tableau)
Optimized $\mathcal{ALC}$-tableaux use the same rules as $\mathcal{ALC}$-tableau in general but they restrict the way these rules are applied.

1. *For the $\sqcap$ and $\forall R$ rule the new formulas $C_1(c)$, $C_2(c)$ $C(d)$ are only added if they do not already occur on the branch.*

2. *For the $\sqcup$ rule: if one of the formulas $C_1(c)$ or $C_2(c)$ already occurs on the branch no action is taken. Only if both of them are not yet present the tableau forks as described by the rule.*

3. *For the $\exists R$ rule: if there already exists a constant such that $R(c, d)$ and $C(d)$ occur on the branch no action is taken. Only if this is not the case both formulas are added with a new constant d.*

We should observe that for every closed tableau there is a closed optimized tabelau with the same root label. Or alternatively, one could check that the soundness and completeness proofs are still true for optimized tableaux. Both claims can be easily seen.

Lemma 78 (Termination Bounds)
Let T be an optimized tableau with root label $C_0(c_0)$. For the purposes of this proof we distinguish the constants that are introduced during the tableau proof from constants that may already be present otherwise. We call the new constants parameters. *In particular c_0 is a parameter. For every branch B of T the following is true*

1. *For every parameter $c \neq c_0$ occuring in B there is a unique sequence of roles $R_1, \ldots R_k$ with $k \geq 1$ and a unique sequence of parameter symbols $c_1, \ldots c_{k-1}$ such that each formula $R_i(c_{i-1}, c_i)$ and $R(c_{k-1}, c)$ occurs on B for $1 \leq i \leq k$.*
   In this case we say that c is a parameter of level k in B.

2. *If a concept expression $C(c)$ occurs on B with c a parameter at level k then $rd(C) \leq rd(C_0) - k$.*
   Thus for any parameter c its level k satisfies $k \leq rd(C_0)$.

3. *If $C(d)$ occurs in B then C is a subexpression of C_0, in symbols $C \in SubEx(C_0)$.*

4. *If $R(c, c_1), \ldots R(c, c_m)$ occur on B for different symbols $c_1, \ldots, c_m$ then m is less than or equal to the number of existential expressions in $SubEx(C_0)$.*

Here rd is the role depth defined in Definition 62 on page 117.

Proof The proof proceeds by structural induction on T. For the initial tableau consisting only of the root node all 4 claims are trivially true.
Consider the case that branch B satisfies all four claims and branch B_1 arises from an application of the $\sqcap$ rule on $C(d) = (C_1 \sqcap C_2)(d) \in B$. Since no new $R(x, y)$ formula is added claims 1 and 4 remain true. For claim two we need only consider the new formulas $C_1(d)$ and $C_2(d)$ and $rd(C_i) \leq rd(C)$ and the induction hypothesis $rd(C) \leq rd(C_0) - k$ yield the claim. For claim 3 we note that C_i is a subexpression of C which in turn is a subexpression of C_0. The case that B_1 arises from B by an application of the $\sqcup$ rule is similar and we skip it.
Let us now consider the more interesting case that B_1 is obtained from B by an application of the $\exists R$ rule on the formula $C(d) = (\exists R.C_1)(d)$. A new parameter d_1 is added and $R(d, d_1)$ is the only R-formula on B involving d_1. Since there is by induction hypothesis a unique chain of role symbols and parameters testifying that d is of level k, there is a unique testifying chain that d_1 is of level $k + 1$. Thus claim 1 is still true. Note also, that the newly added R-formula cannot destroy the previous uniqueness properties. For claim 2 we need to consider the newly added formula $C_1(d_1)$. From the induction hypthesis $rd(C) \leq rd(C_0) - k$ and $rd(C) = rd(C_1) + 1$ we obtain $rd(C_1) = rd(C) - 1 \leq rd(C_0) - (k + 1)$. Claim 3 follows from the obvious observation that C_1 is a subexpression of C which by induction hypothesis is a subsexpression of C_0. We now turn to claim 4. This does not depend on the inductive proof. Let $R(d, d_2), \ldots R(d, d_m)$ be all R-formulas with first parameter d that occur already on B. Now, $R(d, d_1)$ is added. We observe that on each branch of an optimized tableau the $\exists R$ rule can be applied only once to a formula $(\exists R.C_i)(d)$. The only way formulas of the form $R(d, x)$ can surface on a branch is through the application of a rule on $\exists R.C(d)$. The $R(d, d_i)$ thus must arise from different existential formulas.
Finally we consider the case that B_1 is obtained from B by an application of the $\forall R$ rule on the formula $C(d) = (\forall R.C_1)(d)$. The formula $C_1(d')$ is added and we know that $R(d, d')$ occurs already on B. Since no R-formula is added claims 1 and 4 remain trivially true. If k is the level of parameter d then d' is of level $k + 1$. Thus $rd(C_1) = rd(C) - 1 \leq rd(C_0) - (k + 1)$ as required. Finally claim 3 is obviously satisfied since C_1 is a subexpression of C.

∎

Lemma 79 (Termination Lemma)

The construction of an optimized tableau reaches after finitely many steps a tableau were no more rules can be applied.

Proof Let $C(c_0)$ be the root label, s the number of all subexpressions of C_0 and e the number of existential subexpressions. Then at most $e^{rd(C_0)}$ new parameters can be generated. Thus at most $s \times e^{rd(C_0)}$ concept formulas and at most $e^{rd(C_0)}$ R-formulas can occur.
This is a very rough estimate. It is indeed known that the $\mathcal{ALC}$ satisfiability problem is PSPACE complete, which is only slightly better.

▪

Theorem 80 (Tabelau Resoning for A-boxes)
Let $\mathcal{A}$ be an arbitrary, finite A-box.
Let T_0 be the initial tableau that consists of one branch labeled by the formulas in $\mathcal{A}$.

The optimized tableau construction starting with T_0 terminates after finitely many step.

The final tableau is closed if and only if $\mathcal{A}$ is inconsistent.

Proof The proofs are direct consequences from previous theorems and lemmas.

Description Logic with Role Hierarchies

It became soon apparent that for practical purposes the logic $\mathcal{ALC}$ is not expressive enough. Of the many extensions that have been proposed we will consider the description logic called $\mathcal{SHIQ}$.

Definition 76 (Vocabulary)
A vocabulary $\mathbf{V} = \mathbf{C} \cup \mathbf{R}$ for the logic $\mathcal{SHIQ}$ consists of a set $\mathbf{C}$ of concept symbols and a set $\mathbf{R}$ of role symbols such that

1. $\mathbf{C}$ *always contains the symbol $\top$ for the universal concept*

2. *There is a set* $\mathbf{R}^0$ *of atomic roles and for every role symbol* R *there is the symbol* R^- *for the inverse of* R *such that*

$$\mathbf{R} = \mathbf{R}^0 \cup \{R^- \mid R \in \mathbf{R}^0\}$$

3. *There is a subset* $\mathbf{R}_t^0 \subseteq \mathbf{R}^0$ *of transitive atomic roles.*
We say that a role R *is* transitive *if*

   - *either* R *is atomic and* $R \in \mathbf{R}_t^0$
   - *or* $R = R_1^-$ *and* $R_1 \in \mathbf{R}_t^0$

Note that it does not make sense to consider role descriptors of the form $(R^-)^-$ since $(R^-)^- = R$.

Definition 77 (Vocabulary)
A role hierarchy $\mathcal{H}$ *is a finite set of formulas of the form* $R_1 \sqsubseteq R_2$ *for role symbols* $R_1, R_2 \in \mathbf{R}$.
We say that R_1 *is a* subrole *of* R_2.

Definition 78 ($\mathcal{SHIQ}$ expressions)
The set of $\mathcal{SHIQ}$ *expressions of a given vocabulary* $\mathbf{V} = \mathbf{C} \cup \mathbf{R}$ *and a given role hierarchy* $\mathcal{H}$ *is inductively defined by*

1. *every concept symbol* $C \in \mathbf{C}$ *is a* $\mathcal{SHIQ}$ *expression,*
in particular $\top$ *is a* $\mathcal{SHIQ}$ *expression,*

2. *if* $C_1, \ldots, C_k$ *are* $\mathcal{SHIQ}$ *expressions, so are* $C_1 \sqcap \ldots \sqcap C_k$, $C_1 \sqcup \ldots \sqcup C_k$ *and* $\neg C_1$,

3. *if* C *is a* $\mathcal{SHIQ}$ *expression,* R *a role symbol from* $\mathbf{R}$, *then* $\exists R.C$ *und* $\forall R.C$ *are also* $\mathcal{SHIQ}$ *expressions,*

4. *if* R *is a* simple role *in* $\mathbf{R}$, C *a* $\mathcal{SHIQ}$ *expression and* $n \in \mathbb{N}$ *then also* $\leq nR.C$ *and* $\geq nR.C$ *are* $\mathcal{SHIQ}$ *expressions. A role symbol* R *is called* simple *if* R *is not transitive and furthermore that there is no transitive subrole of* R.

Example 22
In addition to the concepts and roles from Example 19 we use here the concept F of all female persons. The expression

$$C = Pa \sqcap \geq 2R.F$$

denotes the set of parents with at least two daughters.

Note, that it is the reference to subroles in clause 4 that makes Definition 78 dependent on the role hierarchy $\mathcal{H}$.

Definition 79 (Interpretation for $\mathcal{SHIQ}$)
An interpretation for the vocabulary $\mathbf{V}$ and a role hierarchy $\mathcal{H}$ consists of a domain of objects Δ and a mapping $\mathcal{I}$ such that:

1. *for every concept symbol $C \in \mathbf{C}$ its interpretation $C^{\mathcal{I}}$ is a subset of Δ, in particular $\top^{\mathcal{I}} = \Delta$,*
2. *for every atomic role symbol $R \in \mathbf{R}^0$ its interpretation $R^{\mathcal{I}}$ is a binary relation on Δ, i.e. $R^{\mathcal{I}} \subseteq \Delta \times \Delta$,*
3. *for every atomic role symbol $R \in \mathbf{R}^0_t$ its interpretation $R^{\mathcal{I}}$ is a transitive relation on Δ,*
4. *for every atomic role symbol $R \in \mathbf{R}^0$ the interpretation of its inverse symbol $(R^-)^{\mathcal{I}}$ is the relation inverse to $R^{\mathcal{I}}$, i.e.*

$$(d_1, d_2) \in (R^-)^{\mathcal{I}} \Leftrightarrow (d_2, d_1) \in R^{\mathcal{I}}$$

5. *if $R_1 \sqsubseteq R_2$ is in $\mathcal{H}$ then $R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$.*

Definition 80 (Semantics of $\mathcal{SHIQ}$ expressions)
Let $\mathcal{I}$ be an interpretation. The meaning of $\mathcal{SHIQ}$ expressions not already covered by Definition 65 is explained as follows

1. $(\leq nR.C)^{\mathcal{I}} = \{d \in \Delta \mid \#\{e \mid (d, e) \in R^{\mathcal{I}} \text{ und } e \in C^{\mathcal{I}}\} \leq n\}$
2. $(\geq nR.C)^{\mathcal{I}} = \{d \in \Delta \mid \#\{e \mid (d, e) \in R^{\mathcal{I}} \text{ und } e \in C^{\mathcal{I}}\} \geq n\}$

Definition 81 (Satisfiability)
Let $\mathcal{H}$ by a role hierarchy.

1. *A set $\mathcal{T}$ of formulas of the form $C_1 \sqsubseteq C_2$ with $\mathcal{SHIQ}$ expressions C_1, C_2 is called a* terminology
2. *An interpretation $\mathcal{I}$ is called a* model *of a terminology $\mathcal{T}$, if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ for all $C_1 \sqsubseteq C_2 \in \mathcal{T}$.*
3. *A $\mathcal{SHIQ}$ expression C is satisfiable with respect to $\mathcal{T}$, if there is a model $\mathcal{I}$ of $\mathcal{T}$ with $C^{\mathcal{I}} \neq \emptyset$*
4. *D subsumes C with respect to $\mathcal{T}$, if for every model $\mathcal{I}$ of $\mathcal{T}$ also $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ is true.*

Theorem 81
The satisfiability and the subsumption problems for $\mathcal{SHIQ}$ are decidable.

Proof see [Horrocks *et al.*, 2000].

If in clause 4 of Definition 78 the restriction to simple roles is dropped the satisfiability problem becomes undecidable. Even if only cardinality formulas of the form $\leq nR.\top$ und $\geq nR.\top$ are allowed, see again [Horrocks *et al.*, 2000].

Adding conjunction of roles to $\mathcal{SHIQ}$ one obtains $\mathcal{SHIQ}^{\sqcap}$. Satisfiability for $\mathcal{SHIQ}^{\sqcap}$ is still decidable but with greater complexity, see [Glimm *et al.*, 2008].

3.7 Knowledge Representation in the Semantic Web

The slogan *Semantic Web* refers to an endeavor to add additional information to resources in the world wide web. The World Wide Web Consortium (W3C) has issued a number of standards for knowledge representation languages that may be used for this purpose. We will present here the Resource Description Framework (RDF) and Web Ontology Language (OWL). We draw on [Allemang & Hendler, 2008, Hitzler *et al.*, 2009] and http://www.w3.org/TR/rdf-mt/ as references.

RDF

The basic concept of the resource description frame work is the *triple*

Definition 82 (RDF Triples)
Let R *be a set of* resources *and* V *a set of* literals*, with* $R \cap V = \emptyset$.
Then every term

$$\langle a, b, c \rangle$$

with $a, b \in R$ *and* $c \in R \cup V$ *is called a triple.*
a *is the* subject*,* b *is the* predicate*, and* c *is called the* object *of the triple.*

In the context of the semantic web everything is either a value or a resource. In other contexts one might have used the word *object*, *entity* or just *thing* instead of resource. But, it does make sense to name the creatures populating the universe of the web *resources*. *Literal* we might think of as strings, numbers and the like. Here is an example

Example 23
Let $R = \{DC20030602, T, D, d, F, L, P\}$ *and* $V = \{st_1, st_2, st_3, st_4, en, st_6\}$.
Then

$$\begin{array}{ll} \langle DC20030602, T, st_1 \rangle & \langle DC20030602, D, st_2 \rangle \\ \langle DC20030602, d, st_3 \rangle & \langle DC20030602, F, st_4 \rangle \\ \langle DC20030602, L, en \rangle & \langle DC20030602, P, st_6 \rangle \end{array}$$

is a set of triples. This is not very illuminating. So, let us be a little less cryptic:

$\langle DC20030602, title,$
"Dublin Core Metadata Element Set, Version 1.1: Reference Description"$\rangle$
$\langle DC20030602, description,$
"The reference description, version 1.1 of the Dublin Core Metadata Element Set"$\rangle$
$\langle DC20030602, date, 2004-12-20\rangle$
$\langle DC20030602, format,$ *"text/html"*$\rangle$
$\langle DC20030602, language, en\rangle$
$\langle DC20030602, publisher,$ *"Dublin Core Metadata Initiative"*$\rangle$

Now, it is clear that we are looking at RDF meta-data for a document, here called DC20030602. The literals $st_1 - st_6$ *have been replaced by their specific (string) values.*

In the logical approach the vocabulary used to build up formulas is fixed locally for a specific paper, or for a whole book by its authors or it is agreed upon by a standard committee. The approach with RDF is different: vocabularies are built and shared globally by the world wide web community. To facilitate this approach resources are given by *Uniform Resource Identifiers* (URI). So, instead of *DC20030602* we should have used http://dublincore.org/documents/2003/06/02/dces/.

Also *title* is a resource with URI http://dublincore.org/2008/01/14/dcelements.rdf#title. Here are a few more correspondences:

Short name	name	URI
	DC20030602	http://dublincore.org/documents/2003/06/02/dces/
T	title	http://dublincore.org/2008/01/14/dcelements.rdf#title
D	description	http://dublincore.org/2008/01/14/dcelements.rdf#description
d	date	http://dublincore.org/2008/01/14/dcelements.rdf#date

A URI can be a URL, for example you can visit the page http://dublincore.org/documents/2003/06/02/dces/. The other URIs including

the # sign all point to the same page, which you can visit likewise. There are precise rules about how URIs are formed set out in the RDF standard and there are agreed ways to abbreviate them via what are called *qnames*. For working with RDF triples these are indispensible but we do not go into these details here. Also the way we represented triple via the $\langle a, b, c\rangle$ notation is our own invention. In the semantic web community it is customary to use XML representations, as exemplified in Figure 3.7.

```
<rdf:RDF>
<rdf:Description rdf:about="http://dublincore.org/documents/
                            2003/06/02/dces/">
<dc:title>
Dublin Core Metadata Element Set, Version 1.1: Reference Description
</dc:title>
<dc:description>
The reference description, version 1.1 of the Dublin
Core Metadata Element Set.
</dc:description>
<dc:date>2004-12-20</dc:date>
<dc:format>text/html</dc:format>
<dc:language>en</dc:language>
<dc:publisher>Dublin Core Metadata Initiative</dc:publisher>
</rdf:Description>
</rdf:RDF>
```

Figure 3.7: XML representation of of meta-data for DC20030602

In addition to the triple

$$\langle DC20030602, title, st_1\rangle$$

we have seen above we also

$$\langle title, issued, 2008\text{-}01\text{-}14\rangle$$

(e.g. on page http://dublincore.org/2008/01/14/dcterms.rdf#title).

The resource *title* may thus occur in the predicate position and also in the subject position, as well – we have not presented an example for this – in

the object position. The same resource may by used in all three positions of a triple. The only restriction is that values may only occur in the third position.

There are two problems that may occur when there is no central authority that fixes the vocabulary.

1. The same name may be used for different things.
2. Different names may be used to denote the same thing.

When URIs are used to define vocabularies the hierarchic organisation of URIs greatly helps in avoiding problem 1. Of course when I set up a vocabulary V_{my} that refers to a web location over which I have control I have to take care that no clashes occur locally in V_{my}. On the other hand it is hard to avoid problem 2. See [Hitzler *et al.*, 2009, Section 2.2.6] for further comments on this naming issue.

So far the choice of a vocabulary was arbitrary and left completely to the user. In the languages RDFS, which stands for RDF with *schemata*, a couple of fixed ressources are assumed to be present in the signature and some constraints are assumed for the semantics of these symbols. In the usual language description first simple triples are considered, then RDF and then RDFS. After our cursory introduction above to simple triples we immediately jump to RDFS. We will consider the following vocabulary

```
rdf:type  rdfs:Class
rdf:Property rdfs:domain rdfs:range
rdfs:subClassOf rdfs:subPropertyOf
```

In the RDFS standard some more symbols are specified. For the reader's information here is a list of those that we skip in our presentation:

```
rdf:XMLLiteral rdf:nil rdf:List rdf:first rdf:rest
rdf:Statement rdf:subject rdf:predicate rdf:object
rdf:Seq rdf:Bag rdf:Alt rdf:_1 rdf:_2 ... rdf:value

rdfs:Resource rdfs:Literal rdfs:Datatype
rdfs:member rdfs:Container rdfs:ContainerMembershipProperty
rdfs:comment rdfs:seeAlso rdfs:isDefinedBy rdfs:label
```

```
 1 rdf:type rdf:type rdf:Property .
 2 rdfs:domain rdfs:domain rdf:Property .
 3 rdfs:range rdfs:domain rdf:Property .
 4 rdfs:subPropertyOf rdfs:domain rdf:Property .
 5 rdfs:subClassOf rdfs:domain rdfs:Class .

 6 rdf:type rdfs:range rdfs:Class .
 7 rdfs:domain rdfs:range rdfs:Class .
 8 rdfs:range rdfs:range rdfs:Class .
 9 rdfs:subPropertyOf rdfs:range rdf:Property .
10 rdfs:subClassOf rdfs:range rdfs:Class .
```

Figure 3.8: Axiomatic Tripels for RFDS

We could now go on and present the restrictions on the semantics of the RDFS symbols listed above. We will take a different approach and first translate triples into atomic formulas in a first order logic. The semantic restrictions will then be given in terms of formulas of first-order logic.

Relation to First-Order Semantics

For the purposes of this section we will consider triples without literals. From a logical point of view literals do not add anything to the understanding of the phenomena we want to address. Also the details of the use of literals in RDFS are quite cumbersome and have been expertly explained elsewhere, see [Hitzler *et al.*, 2009].

Let us compare what we have seen so far with first-order logic. We notice that there are no variables, and therefore also no quantifieres, there are no propositional connectives, except the implicit conjunction - a set of triples is thought of as the conjunction of the triples in the set. So it only makes sense to compare RDF triples with ground atomic formulas in first-order logic. One way to view triples as first-order logic atomic formulas, is to consider a vocabulary V_3 with just one ternary relation symbol and an arbitrary number of constant symbols, see also Exercise 1.2.4. That will certainly do. Another possibility is to consider a vocabulary V_2 which contains a binary relations symbol R_b for every resource b occuring in the predicate position of at least

one triple and the constant symbol b for all resources b occuring in the other two positions. We might then have of course both a constant b and a binary relation R_b at the same time. But, this causes no harm. This basically mimics the normative semantics described in [Hayes, 2004]. We say basically, because the cited reference also explains the interpretation of literals, which we did not go into.

One might have wondered what happened to unary predicates. How do we express in triple notation that a resource x is red? The solution in RDFS is to use the predicate *type* and write $\langle x, \text{type}, \text{red}\rangle$.

We will define the semantics of RDFS by reduction to the semantics of first-order logic.

Definition 83 (First translation into first-order logic)
Let G be a set of triples in the vocabulary V.

The first-order vocabulary V_1 is obtained by treating all symbols in V as constant symbols and adding in addition a binary relation symbol R_b for every b in V. For the rest of this subsection we will uise the shorthand

- *in instead of $R_{rdfs:type}$,*
- *subC instead of $R_{rdfs:subClassOf}$ and*
- *subP instead of $R_{rdfs:subPropertyOf}$.*

Let G^* be the union of G with the axiomatic triples from Figure 3.8.

Let

$$
\begin{array}{lcll}
t_1(G) & = & \{R_p(a,b) \mid \langle a,p,b\rangle \in G^*\} & \\
t_2(G) & = & \{\forall u,v(R_a(u,v) \rightarrow in(u,b) \mid \langle a, rdfs{:}domain, b\rangle \in G^*\} & rdfs2 \\
t_3(G) & = & \{\forall u,v(R_a(u,v) \rightarrow in(v,b) \mid \langle a, rdfs{:}range, b\rangle \in G^*\} & rdfs3 \\
t_4(G) & = & \{\forall u(in(u,a) \rightarrow in(u,b) \mid \langle a, rdfs{:}subClassOf, b\rangle \in G^*\} & rdfs9 \\
t_5(G) & = & \{\forall u,v(R_a(u,v) \rightarrow R_b(u,v) \mid \langle a, rdfs{:}subPropertyOf, b\rangle \in G^*\} & rdfs7 \\
t_6 & = & \{\forall x(in(x, rdfs{:}Property) \rightarrow subP(x,x)), & rdfs6 \\
 & & \quad \forall x,y,z((subP(x,y) \wedge subP(y,z) \rightarrow subP(x,z)), & rdfs5 \\
 & & \quad \forall x(in(x, rdfs{:}Class) \rightarrow subC(x,x)), & rdfs10 \\
 & & \quad \forall x,y,z((subC(x,y) \wedge subC(y,z) \rightarrow subC(x,z))\} & rdfs11
\end{array}
$$

$t(G) = t_1(G) \cup \ldots \cup t_5(G) \cup t_6$

The rdfsi *in the last column are the names of the inference rule used in the W3 standard* `http://www.w3.org/TR/rdf-mt/`.

Definition 84 (RDF Inference)
Let G be a set of triples and $\langle a, p, b\rangle$ an individual triple. We say that $\langle a, p, b\rangle$ can be derived from G, in symbols

$$G \vdash_{RDFS} \langle a, p, b\rangle$$

iff the first-order inference

$$t(G) \vdash R_p(a, b)$$

holds true. In particular $\langle a, p, b\rangle$ is an RDFS tautology if $t(\emptyset) \vdash R_p(a, b)$ holds.

Definition 84 is accompanied by the implicit claim that $G \vdash_{RDFS} \langle a, p, b\rangle$ as defined there coincides with the inference defined in the W3 Standard.

1. in(rdf:type,rdf:Property)
2. R_{domain}(rdfs:domain,rdf:Property)
3. R_{domain}(rdfs:range,rdf:Property)
4. R_{domain}(rdfs:subPropertyOf;rdf:Property)
5. R_{domain}(rdfs:subClassOf,rdfs:Class)
6. R_{range}(rdf:type,rdfs:Class)
7. R_{range}(rdfs:domain,rdfs:Class)
8. R_{range}(rdfs:range,rdfs:Class)
9. R_{range}(rdfs:subPropertyOf,rdf:Property)
10. R_{range}(rdfs:subClassOf,rdfs:Class)

Figure 3.9: Translated Axiomatic Triples for RFDS

Lemma 82 *The following formulas are RDFS tautologies, i.e., they are logical consequences of* $t(\emptyset)$.

1. *in(rdfs:Class,rdfs:Class)*
2. *in(rdf:Property,rdfs:Class)*
3. *in(rdfs:domain;rdf:Property)*
4. *in(rdfs:range,rdf:Property)*
5. *in(rdfs:subPropertyOf,rdf:Property)*
6. *in(rdfs:subClassOf,rdf:Property)*

Proof of 1 Since `rdfs:range rdfs:range rdfs:Class` is an axiomatic triple in $\emptyset^*$, see Figure 3.8(8), we get from Definition 83

$$\forall u, v(R_{\text{range}}(u, v) \rightarrow in(v, \text{rdfs:Class})) \in t_3(\emptyset). \tag{3.11}$$

The fact `rdfs:range rdfs:range rdfs:Class` $\in \emptyset^*$ in addition entails R_{range}(rdfs:range,rdfs:Class) $\in t_1(\emptyset)$. We now obtain from 3.11, as desired in(rdf:Class,rdfs:Class).

Proof of 2 By Figure 3.8(9) we have
`rdfs:subPropertyOf rdfs:range rdfs:Property` $\in \emptyset^*$.
Thus R_{range}(rdfs:subPropertyOf,rdfs:Property) $\in t_1(\emptyset)$. Again using implication 3.11 we arrive at in(rdf:Property,rdfs:Class).

Proof of 3 By Figure 3.8(2) `rdfs:domain rdfs:domain rdfs:Property` $\in \emptyset^*$. Thus by Definition 83

$$\forall u, v(R_{\text{domain}}(u, v) \rightarrow in(u, \text{rdfs:Property})) \in t_2(\emptyset). \tag{3.12}$$

Since also R_{domain}(rdfs:domain,rdfs:Property) $\in t_1(\emptyset)$ we infer by instantiating the universally quantified variables u with rdfs:domain and v with rdfs:Property and modus ponens in(rdfs:domain,rdfs:Property).

Proof of 4 By Figure 3.8(3) `rdfs:domain rdfs:range rdfs:Property` $\in \emptyset^*$, i.e., R_{domain}(rdfs:range,rdfs:Property) $\in t_1(\emptyset)$. Now, 3.12 immediately yields in(rdfs:range,rdfs:Property).

Proof of 5 By Figure 3.8(4)
`rdfs:domain rdfs:subPropertyOf rdfs:Property` $\in \emptyset^*$. By Definition 83 R_{domain}(rdfs:SubPropOf,rdfs:Property) $\in t_1(\emptyset)$. Again using 3.12 yields the desired conclusion in(rdfs:subPropertyOf,rdfs:Property).

Proof of 6 Again we follow the line of argument as in the last subproofs. By Figure 3.8(5) `rdfs:domain rdfs:subClassOf rdfs:Class` $\in \emptyset^*$, i.e., R_{domain}(rdfs:SubClassOf,rdfs:Class) $\in t_1(\emptyset)$ and in a last step 3.12 entails in(rdfs:subClassOf,rdfs:Property).

∎

The attentive reader may have noticed that so far we did not speak about consistency or consistent sets of triples. The plain reason is that this notion does not exist. There is no negation sign in the language of RDFS triples nor any negative requirements in the semantic restrictions. The following triples

```
rdf:type rdf:type rdf:type
rdfs:Class rdfs:Class rdf:Property
rdfs:Class rdfs:Class rdf:Class
rdfs:Class rdfs:SubPropertyOf rdf:domain
rdfs:subClassOf rdfs:domain rdfs:Class
rdf:range rdfs:type rdfs:Class
rdfs:range rdfs:type rdfs:Property
```

are individually and collectively possible triples, though it is hard, or better impossible, to image what they should mean.

Blank Nodes

We start this subsection with another possibility to translate RDF triples into first-order logic vefore we turn to the subject of blank nodes.

Definition 85 (Second translation into first-order logic)
Let G be a set of triples in the vocabulary V.

The first-order vocabulary V_2 is obtained treating all symbols in V as constant symbols and adding in addition one ternary relation symbol Tr.

Let G^* be the union of G with the axiomatic triples from Figure 3.8.

$$
\begin{array}{lcll}
s_1(G) & = & \{Tr(c,d,e) \mid \langle c,d,e\rangle \in G^*\} & \\
s_2 & = & \forall a,b \forall u,v \quad (Tr(a, \textit{rdfs:domain}, b) \wedge Tr(u,a,v)) & \\
 & & \quad \rightarrow Tr(u, \textit{rdfs:type}, b)) & rdfs2 \\
s_3 & = & \forall a,b \forall u,v \quad (Tr(a, \textit{rdfs:range}, b) \wedge Tr((,u,,)a,v) & \\
 & & \quad \rightarrow Tr(v, \textit{rdfs:type}, b)) & rdfs3 \\
s_4 & = & \forall a,b \forall u \quad (Tr(a, \textit{rdfs:subClassOf}, b) \wedge Tr(u, \textit{rdfs:type}, a) & \\
 & & \quad \rightarrow Tr(u, \textit{rdfs:type}, b)) & rdfs9 \\
s_5 & = & \forall a,b \forall u,v \quad (Tr(a, \textit{rdfs:subPropertyOf}, b) \wedge Tr(u,a,v) & \\
 & & \quad \rightarrow Tr(u,b,v)) & rdfs7 \\
s_6 & = & \{\forall x \quad (Tr(x, \textit{rdfs:type}, \textit{rdfs:Property}) & \\
 & & \quad \rightarrow Tr(x, \textit{rdfs:subPropertyOf}, x)), & rdfs6 \\
 & & \forall x,y,z \quad (Tr(x, \textit{rdfs:subPropertyOf}, y) & \\
 & & \quad \wedge Tr(y, \textit{rdfs:subPropertyOf}, z) & \\
 & & \quad \rightarrow Tr(x, \textit{rdfs:subPropertyOf}, z)) & rdfs5 \\
 & & \forall x \quad (Tr(x, \textit{rdfs:type}, \textit{rdfs:Class}) & \\
 & & \quad \rightarrow Tr(x, \textit{rdfs:subClassOf}, x)) & rdfs10 \\
 & & \forall x,y,z \quad (Tr(x, \textit{rdfs:subClassOf}, y) & \\
 & & \quad \wedge Tr(y, \textit{rdfs:subClassOf}, z) & \\
 & & \quad \rightarrow Tr(x, \textit{rdfs:subClassOf}, z))\} & rdfs11
\end{array}
$$

Note that $s_1(G)$ and s_6 are sets, while the remaining s_i are single formulas. Finally

$$s(G) = s_1(G) \cup \{s_2, \ldots s_5\} \cup s_6$$

Again the rdfsi *in the last column are the names of the inference rule used in the W3 standard.*

We leave the topic of translation into first-order logic for a moment and extend the triple representation. So far we have considered sets of RDF triples just as a collection of triples that are all assumed to be true. For example:

⟨z1, line of, 1982953 ⟩ ⟨z2, line of, 1982953 ⟩ ⟨z1, item, Fleece⟩
⟨z1, size, M⟩ ⟨z2, item, Shirt⟩ ⟨z2, size, L⟩

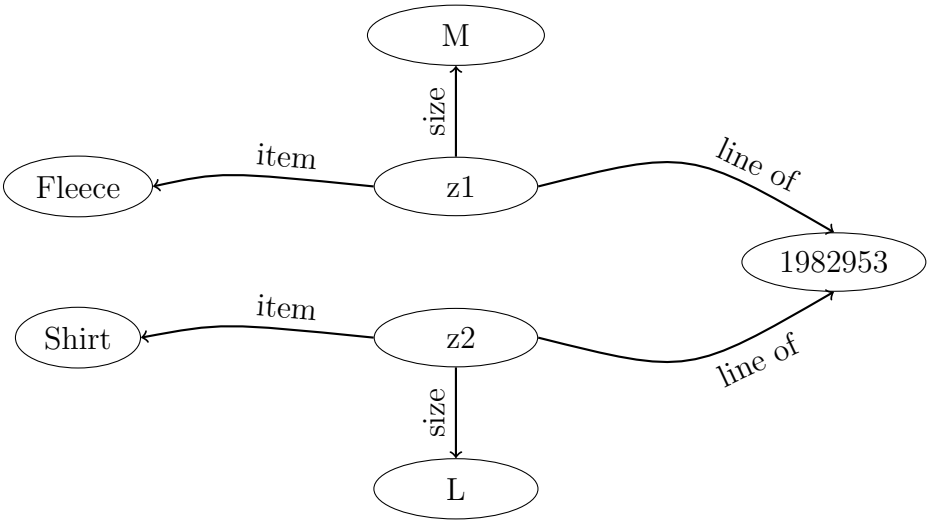


Figure 3.10: A simple RDF graph

It is quite common to arrange triples into a graph, socalled RDF graphs. Subjects and objects are represented as nodes while predicates occur as annotations on edges. The above six triples will result in the graph shown in Figure 3.10.

It is obviously easy to switch from a graphical to a sequential notation. This will become a little bit more complicated when we come to allow blank nodes. The RDF graph in Figure 3.10 results as a part of the solution to Excercise 3.9.23. The resource names z1 and z2 have been introduced to code n-ary relations as conjucntions of binary relations. They do not really matter. For this reason RDF offers the possibility to leave nodes empty as in Figure 3.11. Note that blank nodes can only occur at subject and object positions. There is no such thing as a blank edge label.

A direct translation of the graph in Figure 3.11 would result in

⟨ , line of, 1982953 ⟩ ⟨ , line of, 1982953 ⟩ ⟨ , item, Fleece⟩
⟨ , size, M⟩ ⟨ , item, Shirt⟩ ⟨ , size, L⟩

which obviously does not contain the same information. Thus, to translate

blank nodes into triple notation we have to introduce *place holders* or variables for them, different variables for different blank nodes. We could view z1, z2 in the original triple set as such variables. But, what exactly is the semantics of blank nodes in RDF? We given an answer by extending the translation into first order logic from Definition 85.

Definition 86 (Translation into first-order logic)
Let G be as in Definition 85, but now we assume that G may contain place holders for blank nodes. Let $x_1, \ldots, x_n$ be all such place holders in G. In extending the definition of $S(G)$ to this case we only need to redefine $s_1(G)$:

$$\begin{aligned} s_1(G) \quad = \quad & \{\exists x_1, \ldots, x_n(\textstyle\bigwedge_{\langle c,d,e\rangle \in G} Tr(c,d,e))\} \quad \cup \\ & \{Tr(c,d,e) \mid \textit{ for all axiomatic triples } \langle c,d,e\rangle\} \end{aligned}$$

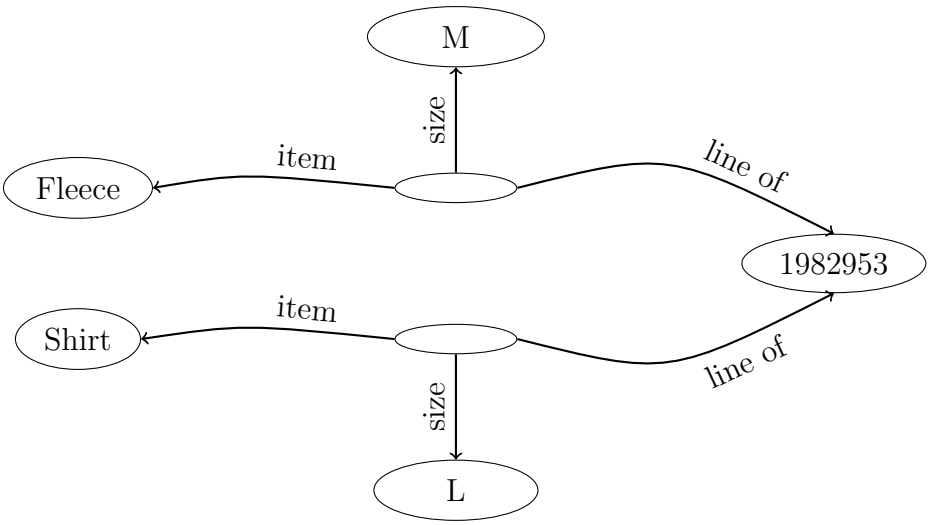


Figure 3.11: A simple RDF graph with blank nodes

OWL(Under Construction)

3.8 Translation into First-Order Logic

In Lemma 66 it was shown that **characterizability** of a frame property can be formalized by formula in **monadic second order logic**. We now want to turn to a simpler problem: formalizing **validity** of modal logic in **first-order logic**. Here is an informal example of what we mean by this. The modal formula $\Diamond p \to \Box p$ is true in a Kripke structure if for every possible world g , such that there is world accessible from g satisfying p then all worlds accessible from g satisfy p. We could formalize the same requirements in first-order logic by $\forall x(\exists y(R(x,y) \wedge p(y)) \to \forall y(R(x,y) \to p(y)))$. At this point of the relationship between the two formalizations is rather informal. The precise definitions and lemmas will follow below. But, we can already see here that two translations are involved: a translation of modal formulas in first-order formulas and a translation of Kripke structures in first-order structures, see Figure 3.12. Furthermore, the formula A^* will contain exactly

$$\begin{array}{ccc} (\mathcal{K}, g) & \models & A \\ \downarrow & & \downarrow \\ \mathcal{K}^* & \models & A^*[g] \end{array}$$

Figure 3.12: Translation from Modal to First-order Formulas

one free variable, denoted by x. We use the more concise notation $\mathcal{M} \models F[g]$ instead of $(\mathcal{M}, \beta) \models F$ with $\beta(x) = g$. One might argue that the translation scheme visualized in Figure 3.12 looks a bit strange. We could think of a weird translation from A to A^* match it with an equally weird translation from $\mathcal{K}$ to $\mathcal{K}^*$ and get a valid diagram. First of all, the translations we will propose below are very intuitve. Second, even if the translations were weird with a bit of extra effort we derive from the diagram: A is a modal tautology iff A^* is a first-order tautology. At this point we no longer care about the details of the translations.

Definition 87 (Formula Translation) *Let* PVar *be a set of propositional variables. The first-order signature* Σ_{PVar} *consists of*

1. *a binary relation* $r(x, y)$

2. *a unary relation* $p(x)$ *for every* $p \in \mathrm{PVar}$.

Every modal formula A *using atoms from* PVar *is inductively translated in a first-order* Σ_{PVar} *formula* A^* *by*

1. $p^* = p(x)$
2. $(A \wedge B)^* = A^* \wedge B^*$, $(A \vee B)^* = A^* \vee B^*$, $(\neg A)^* = \neg A^*$
3. $(\Box A)^* = \forall y(r(x,y) \rightarrow A^*[x/y]$, *with* y *a new variable*
4. $(\Diamond A)^* = \exists y(r(x,y) \wedge A^*[x/y]$, *with* y *a new variable*

The translated formula A^* will contain exactly one free variable, named by x. In the definition of $(\Box A)^*$ we need to replace the subformula A^* by $A^*(\frac{x}{y})$, replacing x by y.

Example 24 ()

$$(\Box\Diamond p \rightarrow \Diamond\Box p)^* = \forall y(r(x,y) \rightarrow \exists z(r(y,z) \wedge p(z))) \rightarrow \exists y(r(x,y) \wedge \forall z(r(y,z) \rightarrow p(z)))$$

`to be done: give more details`

Definition 88 (Structure Translation)
Let $K = (G, R, v)$ *by a Kripke structure. The universe of the translated first-order structure* $\mathcal{K}^*$ *will be* G *and the interpretation of the symbols in* Σ_{PVar} *is given by*

$$\begin{array}{lcl} g \in I(p) & \Leftrightarrow & v(g,p) = \mathbf{true} \\ (g_1, g_2) \in I(r) & \Leftrightarrow & R(g_1, g_2) \end{array}$$

Lemma 83 (Translationlemma)
Let A *be a modal formula,* $\mathcal{K}$ *a Kripke structure* $g \in G$ *then*

$$(\mathcal{K}, g) \models A \text{ iff } \mathcal{K}^* \models A^*[g]$$

Proof: Structural induction on A.
If $A = p$ is an atom, the claim follows directly from Definition 88.
The cases $A = A_1 \wedge A_2$, $A = A_1 \vee A_2$, $A = \neg A_1$ are simple.
If $A = \Box A_1$, then

$$\begin{array}{lll} (\mathcal{K}, g) \models \Box A & \text{iff} & \text{for all } h \text{ with } R(g,h) \text{ we have } (\mathcal{K}, h) \models A \\ & \text{iff} & \text{for all } h \text{ with } R(g,h) \text{ we have } \mathcal{K}^* \models A^*[h] \\ & \text{iff} & \mathcal{K}^* \models \forall y(r(x,y) \rightarrow A^*(\frac{x}{y})[g] \\ & \text{iff} & \mathcal{K}^* \models (\Box A)^*[g] \end{array}$$

The case $A = \Diamond A_1$ proceeds completely analogous.

∎

Theorem 84 (Translationtheorem)
Let A be a modal formula in the vocabulary PVar*. Then*

$$A \text{ is a modal } \mathbf{K} \text{ tautology} \quad \text{iff} \quad \forall x A^* \text{ is a first-order tautology}$$

Proof: This is rather straight forward. Nevertheless, let us spell it out in detail to be prepared for a more complicated situation of the same kind later.

To proof the direction from right to left, we assume that $\forall x A^*$ is a first-order tautology and consider an arbitrary world g in an arbitrary Kripke structure $\mathcal{K}$ with the aim of showing $(\mathcal{K}, g) \models A$. The assumption yields $\mathcal{K}^* \models A^*[g]$ which by Lemma 83 yields $(\mathcal{K}, g) \models A$, as desired.

The reverse implication is a notch more complicated. We assume the left hand side and consider an arbitrary Σ_{PVar} structure $\mathcal{M}$ and an element g in the universe of $\mathcal{M}$. We need to show $\mathcal{M} \models A^*[g]$. It is easy to see that there is a Kripke structure $\mathcal{K}$ with $\mathcal{K}^* = \mathcal{M}$. Since A is a modal tautology we get $(\mathcal{K}, g) \models A$ and Lemma 83 yields the desired conclusion $\mathcal{K}^* \models A^*[g]$.

∎

3.8.1 Decidable Fragments of First-Order Logic

Theorem 85 *For the following classes of formulas the satisfiability problem is decidable:*

1. *All formulas with prefix $\exists^*\forall^*$ with equality and no function symbols (this includes the requirement that there are no constant symbols). Bernays-Schönfinkel-Ramsey class.*

2. *All formulas with prefix $\exists^*\forall^2\exists^*$ no equality, and no function symbols. Gödel-Kalmár-Schütte class.*

3. *All formulas with only 1-place relation and 1-place function symbols. Löb-Gurevich class.*

4. *All formulas with prefix $\exists^*\forall\exists^*$ no equality. Maslov-Orevkov-Gurevich class*

5. *All formulas with prefix $\exists^*$ with equality, no restrictions on relation and function symbols. Gurevich class.*

Proof See [Börger *et al.*, 1982].

Theorem 86 *Let FOL^k denote the fragment of formulas of first-order logic that use at most k different variables.*

1. *FOL^2 is decidable.*

2. *FOL^3 is undecidable.*

Proof For (1) see [Mortimer, 1975] for (2) [Surányi, 1943].

Definition 89 (Guarded Fragment) *The* guarded fragment*, GF, of first-order logic is defined as usual with the exception that quantification is only allowed in the special forms*

$$\forall x(\alpha \rightarrow \phi) \qquad \textit{and} \qquad \exists x(\alpha \wedge \phi)$$

where α is an atomic formula that contains all variables that occur free in ϕ.

Theorem 87 *The guarded fragment GF is decidable.*

Proof See [H.Andréka *et al.*, 1998, Grädel, 1999].

3.9 Exercises

Definition 90
A Kripke frame (G_2, R_2) is a **subframe** *of a Kripke frame (G_1, R_1), if*

1. *$G_2 \subseteq G_1$ and*
2. *for all $g, h \in G_2$ $\quad R_2(g, h)$ iff $R_1(g, h)$*

A subframe (G_2, R_2) of (G_1, R_1) is called a **closed** *subframe, if every world that is accessible from G_2 is already a world in G_2. Formally*

3. *for all $g, h \in G_1$ satisfying $g \in G_2$ and $R_1(g, h)$ we have $h \in G_2$.*

Exercise 3.9.1
Prove the following lemma:
If (G_2, R_2) is a closed subframe of (G_1, R_1), and $v_1 : G_1 \times P \to \{0, 1\}$ an arbitrary interpretation function, and $g \in G_2$ an arbitrary world then for every formula F in $PModFml$:

$$(G_1, R_1, v_1, g) \models F \text{ iff } (G_2, R_2, v_2, g) \models F.$$

Here v_2 is the restriction of v_1 to the domain $G_2 \times P$.

Exercise 3.9.2
Show that the following formulas are **not** *modal tautologies*

1. $(\Box P \to \Box Q) \to \Box(P \to Q)$
2. $\Box(P \vee Q) \to (\Box P \vee \Box Q)$

Exercise 3.9.3
Show that the following two formulas are **S5***-tautologies, i.e. valid in all Kripke structure where the accessability relation R is an equivalence relation.*

1. $\Diamond\Box p \leftrightarrow \Box p$
2. $\Box\Diamond p \leftrightarrow \Diamond p$

Exercise 3.9.4
Prove cases 2,3 and 5 – 11 of Lemma 56.

Exercise 3.9.5
For an arbitrary frame (G, R) prove the following equivalences:

1	$(G,R) \models C(0,1,2,0)$	*iff*	*R ist transitive*
2	$(G,R) \models C(0,1,0,0)$	*iff*	*R ist reflexive*
3	$(G,R) \models C(1,1,0,0)$	*iff*	*R ist symmetric*
4	$(G,R) \models C(1,0,1,0)$	*iff*	*R ist functional*

Exercise 3.9.6
For every $n \in \mathbb{N}$, every Kripke structure $\mathcal{K} = (G, R, v)$ and every world $g \in G$:

1. *$(\mathcal{K}, g) \models \Box^n F$ iff for all $h \in G$ with $R^n(g,h)$ we have $(\mathcal{K}, h) \models F$*
2. *$(\mathcal{K}, g) \models \Diamond^n F$ iff there is $h \in G$ with $R^n(g,h)$ and $(\mathcal{K}, h) \models F$*

Exercise 3.9.7 *Show that the class of non-empty frames, i.e., the class of frames satisfying $\exists x \exists y R(x,y)$ cannot be characterized by a formula in $PModFml$.*
Hint: Use the result of Exercise 3.9.1.

Definition 91
The disjoint sum $(G, R) = (G_1, R_1) \uplus (G_2, R_2)$ of two frames (G_1, R_1), (G_2, R_2) is defined by

1. *$G = G_1 \uplus G_2$*
   If G_1 and G_2 are disjoint G is thus the usual set theoretic union $G_1 \cup G_2$. Otherwise, we need to use disjoint copies of G_1 and G_2, e.g., $G = (\{1\} \times G_1) \cup (\{2\} \times G_2)$
2. *R is the relational sum of R_1 and R_2, i.e.,*
   $R(g,h)$ iff $R_1(g,h)$ or $R_2(g,h)$

Definition 92
The disjoint sum $\mathcal{K} = \mathcal{K}_1 \uplus \mathcal{K}_2$ of two Kripke structures $\mathcal{K}_1 = (G_1, R_1, v_1)$, $\mathcal{K}_2 = (G_2, R_2, v_2)$ is defined by $\mathcal{K} = (G, R, v)$ with

1. $(G, R) = (G_1, R_1) \uplus (G_2, R_2)$

2. $v(g,p) = \begin{cases} v_1(g,p) & \text{if } g \in G_1 \\ v_2(g,p) & \text{if } g \in G_2 \end{cases}$

   This definition works if G_1 and G_2 are disjoint. In the general case we have to take the isomorphisms j_1 and j_2 into account that replace G_i by disjoint copies. Thus

   $$v(g,p) = \begin{cases} v_1(j_1(g),p) & \text{if } g \in G_1 \\ v_2(j_2(g),p) & \text{if } g \in G_2 \end{cases}$$

Exercise 3.9.8 *Assume, for simplicity, that G_1 and G_2 are disjoint sets of worlds and $\mathcal{K}_1 = (G_1, R_1, v_1)$, $\mathcal{K}_2 = (G_2, R_2, v_2)$ Kripke structures and $\mathcal{K} = \mathcal{K}_1 \uplus \mathcal{K}_2$.*

Show that for all $F \in PModFml$:

$$\mathcal{K} \models F \text{ iff } \mathcal{K}_1 \models F \text{ and } \mathcal{K}_2 \models F$$

Exercise 3.9.9 *Show that the class of universal frames, i.e., the class of frames satisfying $\forall x \forall y R(x,y)$ cannot be characterized by a formula in $PModFml$.*
Hint: Use the result of Exercise 3.9.8.

Exercise 3.9.10 *Show that $\Box p \to \Box\Box p$ is a characterizing formula for the class of transitive frames by the method exemplified in Examples 12 and 13.*

Exercise 3.9.11 *Show that $\Box p \to \Diamond p$ is a characterizing formula for the class of serial frames by the method exemplified in Examples 12 and 13.*

Exercise 3.9.12 *Find the characterizing formula for the class of frames of trivial frames $\mathcal{G} = \{(G,R) \mid \neg\exists x \exists y R(x,y)\}$.*
Don't use the Ackermann method for this simple case.

Exercise 3.9.13 *For a Kripke structure $\mathcal{K} = (G, R, v)$, $g \in G$ and a formula $F \in PModFml$ it is intuitive clear that $(\mathcal{K}, g) \models F$ does not depend on worlds $g' \in G$ that can not be reached from g. We will formulate a more detailed statement to the effect that $(\mathcal{K}, g) \models F$ does not depend on world $g' \in G$ that cannot be reached from g within less than $md(F)$ steps (for the definition of the modal depth, $md(F)$, see Definition 35 on page 74).*
To make this a precise statement we need the following definitions.

Definition 93 *Let $\mathcal{K} = (G, R, v)$, $g \in G$. The set $G_g^n \subseteq G$ is defined by induction on n:*

1. $G_g^0 = \{g\}$,
2. $G_g^{n+1} = G_g^n \cup \{g_2 \in G \mid R(g_1, g_2)$ *for some* $g_1 \in G_g^n\}$

We set $\mathcal{K}_s^n = (G_g^n, R_g^n, v_g^n)$ with $R_g^n = R \cap (G_g^n \times G_g^n)$ and $v_G^n = v \cap (G_g^n \times \mathrm{PVar})$.

Claim:
Given $\mathcal{K} = (G, R, v)$, $g \in G$, $n \in \mathbb{N}$ and $F \in PModFml$.
For any $k \leq n$ and $g_1 \in G_g^k$ if $md(F) \leq n - k$:

$$(\mathcal{K}, g_1) \models F \Leftrightarrow (\mathcal{K}_g^n, g_1) \models F$$

For $k = n$, $md(F) \leq n$ this says that $(\mathcal{K}, g) \models F$ is equivalent to $(\mathcal{K}_g^n, g) \models F$. Now, it its your job to prove this.

Exercise 3.9.14 *On page 80 an example was given showing that the deduction theorem is not true for the global modal consequence relation. This exercise presents a substitute.*
For $F_1, F_2 \in PModFml$ with $md(F_1), md(F_2) \leq n$:

$$F_1 \vdash_G F_2 \quad \Leftrightarrow \quad \vdash (F_1 \wedge \Box F_1 \wedge \ldots \Box^n F_1) \rightarrow F_2$$

Exercise 3.9.15 *In the paragraph following Example 19 an example of a translation of description logic expressions into first-order formulas was given.*

1. *Give a general definition of this translation.*
2. *Make precise what is the semantic relationship between the concept expression and its first-order translation.*

Exercise 3.9.16 *Show that the concept formula $\neg\forall R.C = \exists R.\neg C$ is valid.*

Exercise 3.9.17 *Show that if r is a transitive relation on a set Δ then the inverse relation r^- is also transitive.*

Exercise 3.9.18 *Is it possible that a role hierachy $\mathcal{H}$ is inconsistent? i.e., that there is no interpretion $\mathcal{I}$ satisfying $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ for all $C_1 \sqsubseteq C_2$ in $\mathcal{H}$. After all we could have formulas in $\mathcal{H}$ such as $R_1 \sqsubseteq R_2^-$ and $R_2 \sqsubseteq R_1$.*

Exercise 3.9.19 *Show that any non-cyclic T-Box is definitorial.*

Exercise 3.9.20 *Let $\mathcal{K} = (G, R, v)$ be a Kripke structure, $g \in G$. The set of all world reachable from g is defined to be the least subset $RT(g) \subseteq G$ with*

1. *$g \in RT(g)$*
2. *if $h \in RT(g)$ and $R(h, h')$ then $h' \in RT(g)$*

Let $\mathcal{K}^* = (G^*, R^*, v^*)$ with

$$\begin{array}{lcl} G^* & = & RT(g) \\ R^* & = & \text{restriction of } R \text{ to } RT(g) \\ v^* & = & \text{restriction of } v \text{ to } RT(g) \end{array}$$

Then for any modal formula F and any $g* \in RT(g)$:

$$\mathcal{K}, g* \models F \quad \Leftrightarrow \quad \mathcal{K}^*, g* \models F$$

Exercise 3.9.21 *Let A, F be modal formulas then*

$$A \models_G F \quad \Leftrightarrow \quad \{\Box^n A \mid n \geq 0\} \models_L F$$

Exercise 3.9.22 *Show that the translation from $\mathcal{ALC}$ expression into modal logic formulas given in Definition 68 also works in the reverse direction. More precisely, for every modal formula F there is an $\mathcal{ALC}$ expression F° such that the following variation of Lemma 74 is true.*

1. *For every modal formula F and every interpretation $\mathcal{I}$*

$$(F^\circ)^{\mathcal{I}} = \{d \in \Delta \mid (\mathcal{K}_{\mathcal{I}}, g) \models F\}$$

2. *There is an interpretation $\mathcal{I}$ and an object $d \in \Delta$ satisfying $d \in (F^\circ)^{\mathcal{I}}$ iff F is satisfiable.*

Exercise 3.9.23 *The following order may be viewed as an 5-place relation:*

Order: 1982953				
Item	*Id*	*color*	*size*	*quantitiy*
Fleece	*313169M6*	*orange*	*M*	*2*
Shirt	*1911409M6*	*smaragd*	*L*	*1*
Top	*249708M6*	*navy dotted*	*S*	*1*

Translate it into triple notation.

See also Exercise 1.2.5.

Exercise 3.9.24 *Taken from B. Nebels lecture. Translate the following $\mathcal{SHIQ}$ expressions in satisfiability equivalent formulas of first order logic*

1. $\forall r^{-1}.(C \sqcap \neg\exists s.D)$
2. $\forall r \sqcap s.(\forall t.(\neg C \sqcup \exists r.D))$
3. $\exists r.((C \sqcup (\neg s \sqsubseteq t)) \sqcap \forall s.(t \sqsubseteq u))$

Exercise 3.9.25 *Correctness and completeness theorems 69 and 71 were done for the modal logic* **K**. *If we redo the proofs for the modal logic* **T** *(with reflexive accessability relation R) where will it be neccessary to use the definition of* **T***-accessability of prefixes?*

Exercise 3.9.26 *The first-order semantics for the vocabulary V_2 introduced in Definition 85 would require to provide a universe Δ and an interpretation function I such that $I(c) \in \Delta$ for each constant symbol in V_2 and $I(Tr) \subseteq \Delta^3$ for the only relation symbol $Tr \in V_2$.*

The triple semantics given in `http://www.w3.org/TR/rdf-mt/` *also requires a universe Δ and an interpretation function I such that $I(c) \in \Delta$ for each constant symbol in V_2. But now, things differ. The RDF triple semantics requires that there is a function $ext : \Delta \to \mathbb{P}(\Delta^2)$, i.e., the function ext associates with every element $d \in \Delta$ a subset $ext(d) \subseteq \Delta^2$ of pairs of elements from Δ.*

Can you see that these two approaches are equivalent?

Exercise 3.9.27 *C_1, C_2 are concept symbols, R a role symbol. Which of the following inclusions, if any, are universally valid?*

1. $C_1 \sqcap \exists R.C_2 \sqsubseteq \exists R.(C_1 \sqcap C_2)$
2. $\exists R.(C_1 \sqcap C_2) \sqsubseteq C_1 \sqcap \exists R.C_2$

Chapter 4

Dynamic Logic

Dynamic Logic provides a logical framework to describe and reason about complex systems that are determined by a set of states and actions transforming the system from one state to another. This is a very general application area. The systems we will consider here will be programming languages. In this case the states are the states of computation reached during the execution of a computer program, while the actions are the commands of the programming language.

The roots for this type of a logic for programs go back to the paper [V.R.Pratt, 1976] by V. R. Pratt. The name *Dynamic Logic* was coined by D. Harel in [Harel, 1979], which is an extension of his dissertation completed in 1978. The exposition in this paper is for the greatest part based on Harel's work. Other references still worth reading today are [Harel, 1984], [Goldblatt, 1987, Section 10] und, [Goldblatt, 1982, Part Three]. The most recent account is the book [Harel *et al.*, 2000].

4.1 Motivating Example

We will use the well known puzzle called *the towers of Hanoi* to explain the principal approach of dynamic logic. The goal of the game is to move all

(a) Initial State (b) Final State

Figure 4.1: The Towers of Hanoi

disks from the first pile to the last pile making use of an auxiliary pile in the middle, as shown in Figure 4.1. Of course only the top most disk of a pile can be moved. Furthermore, the restriction has to be observed that never a larger disk may be placed on top of a smaller one.

The towers of Hanoi puzzle is mostly used as an example for recursive programming. But, there is also a suprisingly simple iterative solution that can be described by the following instructions:

1. Move alternatingly the smallest disk and another one.

2. If moving the smallest disk put it on the stack it did not come from in its previous move.

3. If not moving the smallest disk do the only legal move,

We could try to describe this sequence of actions by

$$moveS; moveO; moveS; moveO; \ldots$$

Or even more concise if we use the unbounded iteration operator * known from regular expressions:

$$(moveS; moveO)^*$$

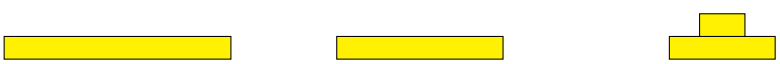
This shows that in Dynamic Logic we can explicitly refer to actions and that it is possible to compose composite actions from simpler ones. What we have is still a very crude approximation of the solution of the towers of Hanoi puzzle. We need to add an action, that checks for termination:

$$moveS; testForStop; (moveO; moveS; testForStop)^*$$

This is a perfect description of the required actions. It could for example be used by the legendary priest in the legendary Indian temple to make one move with the 64 golden disks per day until after he moved the last disk on the rightmost pile the world will come to an end. The description of the required actions is sufficient in a situation where you have access to the real world and are confronted with a physical realisation of the puzzle. If we want to argue and reason about it we also need a formal presentation of the possible states. This is in this case fairly easy. We may e.g., use the following two-place function *stack*:

$$stack(n, m) = \begin{cases} k > 0 & \text{on stack } n \text{ at position } m \\ & \text{there is a disk of size } k \\ 0 & \text{on stack } n \text{ at position } m \\ & \text{there is no disk} \end{cases}$$

with $1 \leq n \leq 3$ and $1 \leq m \leq d$ with d the number of disks. Thus the following state



would be represented by the following functions, which we here define via its function table.

stack	$first$	$second$	$third$
position 4	0	0	0
position 3	0	0	0
position 2	0	0	1
position 1	4	3	2

The final position could then be characterised by the formula

$$\forall m(1 \leq m \leq d \rightarrow stack(3, m) \neq 0)$$

This formula says that all disks are on stack 3. Another property we would want to be satisfied throughout is that no larger disk is placed on top of a smaller one:

$$\begin{array}{ll} \bigwedge_{1\leq n\leq 3} \forall m_1, m_2 & ((1 \leq m_1 < m_2 \leq d \wedge stack(n, m_1) \neq 0) \\ & \rightarrow stack(n, m_1) > stack(n, m_2)) \\ \wedge & \\ \bigwedge_{1\leq n\leq 3} \forall m_1, m_2 & ((1 \leq m_1 < m_2 \leq d) \\ & \rightarrow stack(n, m_1) \geq stack(n, m_2)) \end{array}$$

i.e., on each of the three pegs the disks are arranged in decreasing order. So far we have seen examples of properties of states formalised in first-order predicate logic. It is a particular feature of Dynamic Logic that it allows to relate properties of states to actions. One of the simplest relations of this kind is to demand that a formula ϕ should be an invariant for an action A:

> whenever ϕ is true before A
> it is also true after the execution of action A.

Formally we could write

$$\text{OrderedStacks} \rightarrow \langle moveS \rangle \text{ OrderedStacks}$$

when we agree that a formula of the form $\langle A \rangle \ \phi$ should be read as: after performance of action A a state is reached that satisfies formula ϕ. This already points to the issue of termination; do actions always terminate? We are e.g., not certain that the action $moveS; testForStop; (moveO; moveS; testForStop)^*$ terminates. It could

loop infinitely and never pass through the final state. Thus we do not require that actions terminate. A statement of the form $\langle A \rangle\ \phi$ however incorporates the claim that A terminates. The formula

$$\langle moveS; testForStop; (moveO; moveS; testForStop)^* \rangle \mathbf{true}$$

thus says that $moveS$; $testForStop$; $(moveO; moveS; testForStop)^*$ always terminates. In other cases we might only want to make a partial assertion, if action A terminates it should be in a state statisfying ϕ, if A does not terminate we do not care. For this the formal notation $[A]\ \phi$ would be used.

4.2 Syntax and Semantics of Regular Dynamic Logic

For the definition of the syntax of Dynamic Logic $\mathbf{DL}_{reg}$ we start - as usual - by fixing a vocabulary Σ. We proceed - again as usual - by defining terms DLT_Σ and formulas DLF_Σ of the logic. In addition - and that is particular for Dynamic Logic - we will define the set of programs Π. The definitions of DLF_Σ and Π will be mutually recursive.

Definition 94 (Vocabulary) *A vocabulary Σ consists of*

- *a set of function symbols $f, g, f_i, \ldots$ with fixed number of arguments,*
- *0-place function symbols will also be called constant symbols,*
- *a set of predicate symbols $p, q, p_i, \ldots$ with fixed number of arguments.*

By Var we denote an infinite set of variable symbols.

Definition 95 (DL Terms)

1. $x \in DLTerm_\Sigma$ *for* $x \in$ Var
   Every variable symbols is a term.
2. $f(t_1, \ldots, t_n) \in DLTerm_\Sigma$
   for every n-place function symbol $f \in \Sigma$ *and* $t_1, \ldots, t_n \in DLTerm_\Sigma$

Definition 96 (DL Formulas and Programs)

1. *atomic formulas*
   $r(t_1, \ldots, t_n) \in DLFml_\Sigma$
   for every n-place relation symbol $r \in \Sigma$ *and terms* $t_i \in DLTerm_\Sigma$.

2. *equations*
   $t_1 = t_2 \in DLFml_\Sigma$
   for $t_1, t_2 \in DLTerm_\Sigma$

3. *closure under predicate logic operators*
   $F_1 \vee F_2$ $F_1 \wedge F_2$, $F_1 \to F_2$, $\neg F_1$, $\forall x F_1$ *and* $\exists x F_1 \in DLFml_\Sigma$
   for all $F_1, F_2 \in DLFml_\Sigma$.

4. *modal operators*
   $[\pi]F, \langle\pi\rangle F \in DLFml_\Sigma$
   for $F \in DLFml_\Sigma$ *and* $\pi \in \Pi$.

5. *atomic programs*
   $(x := t) \in \Pi$ *for* $t \in DLTerm_\Sigma$ *and* $x \in \mathrm{Var}$.

6. *composite programs*
   If π_1,$\pi_2 \in \Pi$ *then*

   (a) $\pi_1; \pi_2 \in \Pi$ — *sequential composition*
   (b) $\pi_1 \cup \pi_2 \in \Pi$ — *nondeterministic choice*
   (c) $\pi^* \in \Pi$ — *iteration*

7. *tests*
   $con? \in \Pi$ *for every quantifierfree formula* $con \in DLFml_\Sigma$.

There are other choices for the set Π of programs, see e.g., Section 4.2.1. Our choice is called the set of *regular* programs, because of the close similarity to the syntax of regular expressions. Consequently the particular version of Dynamic Logic is called *regular Dynamic Logic*. Other versions could e..g, be based on the concept of while programs known from theoretical Computer Science, or Π could consist of the programs of a real programming language like Java, see [Beckert *et al.*, 2007, Chapter 3].

Before we give examples let us first define the semantics of regular Dynamic Logic.

Let $\mathcal{M} = (M, val_{\mathcal{M}})$ be a first-order structure. As usual the function $val_{\mathcal{M}}$ interprets function and relation symbols as functions and relations of the universe M. Will will use the same notation $val_{\mathcal{M}}$ for the interpretation of terms and formulas. In general it is necessary to provide in addition an assignments of the (free) variables in terms and formulas. We use $\mathrm{Var} \to M$ to denote the set of variable assignments for $\mathcal{M}$. For $u \in (\mathrm{Var} \to M)$ and $t \in DLTerm_\Sigma$ we will use the notation $val_{\mathcal{M},u}(t)$ for the usual first-order evaluation of t in $\mathcal{M}$ with variables in t interpreted via u. For $s \in (\mathrm{Var} \to M)$, $x \in \mathrm{Var}$, $a \in M$ we will need the following notation

$$s[x/a](y) = \begin{cases} a & \text{if } y = x \\ s(y) & \text{otherwise} \end{cases}$$

Definition 97 (Kripke Structure) *For every given first-order structure $\mathcal{M} = (M, val_{\mathcal{M}})$ the Kripke structure*

$$\mathcal{K}_{\mathcal{M}} = (S, \rho, \models)$$

is determined by

$S = \mathrm{Var} \to M$	*the set of states*
$\rho : \Pi \to S \times S$	*accessibility relations*

subject to the following constraints:

$(s, s') \in \rho(x := t)$	*iff*	$s' = s[x/val_{\mathcal{M},s}(t)]$
$(s, s') \in \rho(\pi_1; \pi_2)$	*iff*	*there exists* $t \in S$ *with* $(s, t) \in \rho(\pi_1)$ *and* $(t, s') \in \rho(\pi_2)$
$(s, s') \in \rho(\pi_1 \cup \pi_2)$	*iff*	$(s, s') \in \rho(\pi_1)$ *or* $(s, s') \in \rho(\pi_2)$
$(s, s') \in \rho(\pi^*)$	*iff*	*there exists* n *and* $s_1, \ldots s_n \in S$ *such that* $s_1 = s$ *and* $s_n = s'$ *and* $(s_i, s_{i+1}) \in \rho(\pi)$ *for* $1 \leq i < n$
$(s, s') \in \rho(con?)$	*iff*	$s = s'$ *and* $s \models con$

$\mathcal{M}$ is called the domain of computation of $\mathcal{K}$.

Let $\mathcal{K}$ be a Kripke structure with computation domain $\mathcal{M} = (M, val_{\mathcal{M}})$. For every formula $F \in DLFml$ and every state s of $\mathcal{K}$ the following definition

will explain inductively when F is true in state s, written as $s \models F$. The meaning $\rho(\pi)$ of a program $\pi \in \Pi$ given as a set of pairs of begin state and end state is already completely determined by the contraints in Definition 97.

Definition 98 (DL_{reg} Semantics)

$s \models r(t_1, \ldots, t_n)$	*iff*	$(val_{\mathcal{M},s}(t_1), \ldots, val_{\mathcal{M},s}(t_n)) \in val_{\mathcal{M}}(r)$
$s \models t_1 = t_2$	*iff*	$val_{\mathcal{M},s}(t_1) = val_{\mathcal{M},s}(t_2)$
$s \models F$		F *matching one of* $F_1 \vee F_2, F_1 \wedge F_2,$ $F_1 \rightarrow F_2, \neg F_1, \forall x F_1$ *or* $\exists x F_1$ *as usual.*
$s \models [\pi]F$	*iff*	$s' \models F$ *for all* s' *with* $(s, s') \in \rho(\pi)$
$s \models \langle\pi\rangle F$	*iff*	*there exists* s' *with* $(s, s') \in \rho(\pi)$ *and* $s' \models F$

If $s \models F$ *holds, we say that* formula F is true in state s. *If it is not clear from the context, we say* F is true in state s of $\mathcal{K}$, *and write in symbols* $(\mathcal{K}, s) \models F$.

We say F is true in the Kripke structure $\mathcal{K}_{\mathcal{M}} = (S, \rho)$, *in symbols* $\mathcal{K}_{\mathcal{M}} \models F$, *if* $s \models F$ *is true for all* $s \in S$.

Notice, that once the domain of computation $\mathcal{M}$ is fixed, the Kripke structure $\mathcal{K}_{\mathcal{M}} = (S, \rho, \models)$ is uniquely determined.

We note the following elementary observations on the semantics of regular programs.

Lemma 88 *For any program* $\pi \in \Pi$ *let* $FV(\pi)$ *be the set of variables occuring on the left hand side of an assignment statement in* π *and* V^π *all variables occuring in* π.

1. *The program* π *only changes variables in* $FV(\pi)$;
   that is, if $(s, s_1) \in \rho(\pi)$ *then* $s(x) = s_1(x)$ *for all variables* $x \notin FV(\pi)$.

2. *Variables outside* V^π *do not influence the program* π;
   that is, if $x \notin V^\pi$ *and* $(s, s_1) \in \rho(\pi)$ *then also*
   $(s[x/a], s_1[x/a]) \in \rho(\pi)$ *for arbitrary* a.

3. *more general: If* $(s, s_1) \in \rho(\pi)$ *and* s' *is a variable assignment such that* $s'(y) = s(y)$ *for all* $y \in V^\pi$ *then there is* s'_1 *such that*

(a) $(s', s'_1) \in \rho(\pi)$ *and*
(b) $s'_1(x) = s'(x)$ *for all* $x \notin V^\pi$
(c) $s'_1(y) = s_1(y)$ *for all* $y \in V^\pi$.

Proof Simple.

∎

Now it is time to look at a few examples.

Example 25 *Let's warm up with a simple question: For a formula con and a program* π *what does* $(u, u') \in \rho(con?; \pi)$ *mean? Here is the answer:*

$$\begin{array}{lll} (u, u') \in \rho(con?; \pi) & \textit{iff} & \textit{exists } w \textit{ with} \\ & & (u, w) \in \rho(con?) \textit{ and } (w, u') \in \rho(\pi) \\ & \textit{iff} & \textit{exists } w \textit{ with} \\ & & u \models con,\ w = u \textit{ and } (w, u') \in \rho(\pi) \\ & \textit{iff} & u \models con \textit{ and } (u, u') \in \rho(\pi) \end{array}$$

Example 26 *We can use Example 25 to find out the meaning of* $(u, u') \in \rho((con?; \pi_1) \cup (\neg con?; \pi_2))$

$$\begin{array}{l} (u, u') \in \rho((con?; \pi_1) \cup (\neg con?; \pi_2)) \\ \quad \textit{iff} \quad (u, u') \in \rho((con?; \pi_1)) \textit{ or } (u, u') \in \rho((\neg con?; \pi_2)) \\ \quad \textit{iff} \quad u \models con \textit{ and } (u, u') \in \rho(\pi_1) \textit{ or } u \models \neg con \textit{ and } (u, u') \in \rho(\pi_2) \\ \quad \textit{iff} \quad (u, u') \in \rho(\textbf{if } con \textbf{ then } \pi_1 \textbf{ else } \pi_2) \end{array}$$

Thus:

$$(con?; \pi_1) \cup (\neg con?; \pi_2) \equiv (\textbf{if } con \textbf{ then } \pi_1 \textbf{ else } \pi_2)$$

Example 27

$$\begin{array}{l} (u, w) \in \rho((A?; \pi)^*; \neg A?) \\ \quad \textit{iff} \quad \textit{there exist } n \in \mathbb{N} \textit{ and } u_1, \ldots, u_n \in S \textit{ with } u_1 = u \\ \qquad (u_i, u_{i+1}) \in \rho(A?; \pi) \textit{ for all } i,\ 1 \leq i < n \textit{ and} \\ \qquad (u_n, w) \in \rho(\neg A?) \\ \quad \textit{iff} \quad \textit{there exist } n \in \mathbb{N} \textit{ and } u_1, \ldots, u_n \in S \textit{ with } u_1 = u,\ u_n = w \\ \qquad (u_i, u_{i+1}) \in \rho(\pi) \textit{ and } u_i \models A \textit{ for all } i,\ 1 \leq i < n \textit{ and} \\ \qquad w \models \neg A \\ \quad \textit{iff} \quad (u, w) \in \rho(\textbf{while } A \textbf{ do } \pi) \end{array}$$

Thus

while A **do** $\pi \quad \equiv \quad (A?;\pi)^*;\neg A?$

Examples 26 and 27 show that the programs covered by regular Dynamic Logic are at least as powerful as while programs.

Next item on the agenda for introducing a logic is the explanation of logical validity and derivability for Dynamic Logic. In first-order logic there is just one natural way to define these concepts. For Dynamic Logic there are choices. First we distinguish the uninterpreted case form the interpreted case. In the uninterpreted case we take into account all structures $\mathcal{M}$ as domains of computation that fit the given vocabulary. This corresponds most closely to universal validity in first-order logic. In the interpreted case we fix one domain of computation. This is most useful in Computer Science applications, where we want e.g., a fixed model of arithmetic. On a second level we distinguish local and global logical consequence relations, as is done in many modal logics.

Definition 99 (Uninterpreted Validity)

$\vdash F$	*F is valid*	$\mathcal{K}_{\mathcal{M}} \models F$ *for all* $\mathcal{M}$.
$G \vdash F$	*G (locally) entails F*	*for all* $\mathcal{M}$ *and all* $s \in S$ *if* $s \models G$ *then also* $s \models F$
$G \vdash^g F$	*G globally entails F*	*for all* $\mathcal{M}$ *if* $s \models G$ *for all* $s \in S$ *then* $s \models F$ *for all* $s \in S$

Definition 100 (Deterministic Programs)
A program π is called deterministic *if for all Kripke structures $\mathcal{K} = (S, \rho, \models)$ $(s, s_1) \in \rho(\pi)$ and $(s, s_2) \in \rho(\pi)$ implies $s_1 = s_2$.*

Lemma 89 (Uninterpreted Tautologies) *Assume that the variable x does not occur in the program π. Then,the following formulas are valid in the uninterpreted semantics.*

1. $(\exists x\ \langle\pi\rangle F) \leftrightarrow (\langle\pi\rangle \exists x\ F)$
2. $(\forall x\ [\pi] F) \leftrightarrow ([\pi] \forall x\ F)$

3. $(\exists x\ [\pi]F) \to ([\pi]\exists x\ F)$

4. $([\pi]\exists x\ F) \to (\exists x\ [\pi]F)$ *if π is deterministic*

5. $(\langle\pi\rangle\forall x\ F) \to (\forall x\ \langle\pi\rangle F)$

6. $(\forall x\ \langle\pi\rangle F) \to (\langle\pi\rangle\forall x\ F)$ *if π is deterministic*

7. $(\langle\pi\rangle(F \wedge G)) \to ((\langle\pi\rangle F) \wedge \langle\pi\rangle G)$

8. $(\langle\pi\rangle(F \wedge G)) \leftrightarrow ((\langle\pi\rangle F) \wedge \langle\pi\rangle G)$ *if π is deterministic*

Proofs

Proof of (1). Since this is the first proof in a series of similar spirit we present it in full detail. The assumption $x \notin V^\pi$ will make the applications of the elementary observations in Lemma 88 in the following proof possible.

$\Rightarrow$:

1	s	$\vDash$	$\exists x\langle\pi\rangle F$	assumption
2	$s[x/b]$	$\vDash$	$\langle\pi\rangle F$	for some b
3	t	$\vDash$	F	with $(s[x/b], t) \in \rho(\pi)$
4	t	$\vDash$	$\exists x F$	classical predicate logic
5	$t[x/s(x)]$	$\vDash$	$\exists x F$	classical predicate logic
6	s	$\vDash$	$\langle\pi\rangle\exists x F$	from $(s[x/b], t) \in \rho(\pi)$ we get by Lemma 88 $(s, t[x/s(x)]) \in \rho(\pi)$

$\Leftarrow$:

1	s	$\vDash$	$\langle\pi\rangle\exists x F$	assumption
2	t	$\vDash$	$\exists x F$	with $(s, t) \in \rho(\pi)$
3	$t[x/b]$	$\vDash$	F	for some b
4	$s[x/b]$	$\vDash$	$\langle\pi\rangle F$	from $(s, t) \in \rho(\pi)$ we get by Lemma 88 $(s[x/b], t[x/b]) \in \rho(\pi)$
5	s	$\vDash$	$\exists x\langle\pi\rangle F$	predicate logic

Proof of (2). Follows from (1) and Exercise 4.7.4(1)

Proof of (3). From now on, we will not prove the two implications separately. Any line in the following proof outlines is an equivalence transformation.

Consider an arbitrary state s in a Kripke structure $\mathcal{K}_{\mathcal{M}}$.

1	s	$\models$	$\exists x\,[\pi]F$	assumption
2	$s[x/a]$	$\models$	$[\pi]F$	for some a
3	t	$\models$	F	for all t with $(s[x/a],t)\in\rho(\pi)$
3	t	$\models$	$\exists xF$	for all t with $(s[x/a],t)\in\rho(\pi)$
4	$s[x/a]$	$\models$	$[\pi]\exists xF$	Def. of [ ] operator
5	s	$\models$	$[\pi]\exists xF$	since $x\notin V^{\pi}$

Proof of (4). Consider again an arbitrary state s in a Kripke structure $\mathcal{K}_{\mathcal{M}}$.

1	s	$\models$	$[\pi]\exists xF$	assumption
2	t	$\models$	$\exists xF$	for all t with $(s,t)\in\rho(\pi)$ by determinacy this is equivalent to:
3	t	$\models$	$\exists xF$	for the unique t with $(s,t)\in\rho(\pi)$
4	$t[x/b]$	$\models$	F	for the unique t with $(s,t)\in\rho(\pi)$ and some $b\in M$
5	$s[x/b]$	$\models$	$[\pi]F$	since by the basic observation $t[x/b]$ is the unique state with $(s[x/b],t[x/b])\in\rho(\pi)$ for some $b\in M$
5	$s[x/b]$	$\models$	$\exists x[\pi]F$	predicate logic
6	s	$\models$	$\exists x[\pi]F$	predicate logic

Proof of (5). Follows from (3) and Exercise 4.7.4(1).

Proof of (6). Follows from (4) and Exercise 4.7.4(2).

Proof of (7).

1	s	$\models$	$\langle\pi\rangle(F\wedge G)$	assumption
2	t	$\models$	$F\wedge G$	with $(s,t)\in\rho(\pi)$
3	t	$\models$	F and $t\models G$	
4	s	$\models$	$\langle\pi\rangle F$ and $s\models\langle\pi\rangle G$	
5	s	$\models$	$\langle\pi\rangle G\wedge\langle\pi\rangle F$	

Proof of (8). Because of (7) only the implication $\leftarrow$ needs to be proved.

1	s	$\models$	$\langle\pi\rangle F \wedge \langle\pi\rangle G$	assumption
2	t_1	$\models$	F and $t_2 \models G$	with $(s, t_i) \in \rho(\pi)$
3	t_2	$\models$	F and $t_2 \models G$	since t_1 and t_2 coincide on $x \notin V^\pi$ and no $x \in V^\pi$ occurs in F
4	t_2	$\models$	$F \wedge G$	
5	s	$\models$	$\langle\pi\rangle(F \wedge G)$	

■

Example 28

Assume that the vocabulary Σ contains the two unary function symbols f, g and a unary predicate symbol p and let $\mathcal{M}$ be an arbitrary structure for Σ. Then the following DL formula is valid in the Kripke structure $\mathcal{K}_\mathcal{M}$.

$$x = y \wedge \forall x(f(g(x)) = x) \rightarrow \\ [\textbf{while } p(y) \textbf{ do } y := g(y)]\langle\textbf{while } y \neq x \textbf{ do } y := f(y)\rangle\textbf{true}$$

Let s be a state of $\mathcal{K}$ satisfying with $s(x) = s(y)$ and $s \models \forall x(f(g(x)) = x)$. We need to show

$$s \models [\textbf{while } p(y) \textbf{ do } y := g(y)]\langle\textbf{while } y \neq x \textbf{ do } y := f(y)\rangle\textbf{true}$$

That is to say, for any t such that $(s, t) \in \rho(\textbf{while } p(y) \textbf{ do } y := g(y))$ we need to show that there is t' such that $(t, t') \in \rho(\textbf{while } y \neq x \textbf{ do } y := f(y))$ By the choice of t we know that there is an $n \in \mathbb{N}$ and there are states $t_1, \ldots, t_n$ with $t_1 = s$ and $t_n = t$ such that $t_i \models p(y)$ for all $1 \leq i < n$, $t_n \models \neg p(y)$ and $t_{i+1}(y) = g(t_i(y))$ for all $1 \leq i < n$, i.e., $t_n(y) = g^{n-1}(s(y))$. Also we know that $s(z) = t_i(z)$ for all $z \neq y$ and all i.
Let u_j $1 \leq j \leq n$ be states such that $u_j(z) = s(z) = t_i(z)$ for all j and $z \neq y$ and $u_j(y) = f^{j-1}(t_n(y))$, in particular that means $u_1(y) = t_n(y)$. Obviously $(u_j, u_{j+1}) \in \rho(y := f(y))$. From $u_n(y) = f^{n-1}(t_n(y)) = f^{n-1}(g^{n-1}(s_n(y)))$ and $\mathcal{M} \models \forall x(f(g(x)) = x)$ we obtain $u_n(y) = s(y)$. By the initial assumption on s and the fact that the value of x is never changed we get $u_n(y) = s(x) = u_n(x)$. This shows that $(t_n, u_n) \in \rho(\textbf{while } y \neq x \textbf{ do } y := f(y))$ and we are finished. Since we have no information on what f and g do the while-loop might terminate earlier, but it will certainly terminate after n iterations.

Definition 101 (Interpreted Validity)

Let $\mathcal{M}$ be a fixed first-order structure.

$\vdash_{\mathcal{M}} F$	*F is $\mathcal{M}$-valid*	$\mathcal{K}_{\mathcal{M}} \models F$.
$G \vdash_{\mathcal{M}} F$	*G (locally) $\mathcal{M}$-entails F*	*for all $s \in S$ if $s \models G$ then also $s \models F$*
$G \vdash^{g}_{\mathcal{M}} F$	*G globally $\mathcal{M}$-entails F*	*if $s \models G$ for all $s \in S$ then $s \models F$ for all $s \in S$*

Example 29

Let $\mathcal{M} = (\mathbb{N}, 0, +, -, >)$ be the computational domain of the natural numbers. The following formulas are $\mathcal{M}$-tautologies.

1. $(p(0) \wedge \forall x(p(x) \rightarrow p(x+1))) \rightarrow \forall x p(x)$
2. $\neg\exists x(0 < x \wedge x < 1)$

Example 30

Fixing a computation domain greatly increases the expressive power. Let $\mathcal{M}$ be the natural numbers as in the previous example 29 and R a binary relation symbol in the vocabulary Σ. Then we can define the transitive closure of R. We start with the auxiliary definition:

$$\begin{aligned} TC^0_R(x,y,z) \quad \leftrightarrow \quad & (z = 0 \wedge x = y) \vee \\ & (z > 0 \wedge \exists u(TC^0_R(x,u,z-1) \wedge R(u,y)) \end{aligned}$$

and arrive at the definition

$$TC_R(x,y) \leftrightarrow \exists z(TC^0_R(x,y,z))$$

It is easily seen that $TC_R(x,y)$ defines the reflexive, transitive closure of R.

This example also shows that Definitions 99 and 101 may be refined to allow part of the vocabulary Σ to stay fixed while the rest may vary.

4.2.1 Boogie PL

As an alternative to the language of regular programs from Definition 96 we sketch in this section the intermediate programming language, sometimes also called a verification language, Boogie PL from [Barnett *et al.*, 2005, Leino, 2008], which plays an important role in Microsoft's program verification tool suite. We leave out some details and concentrate on the parts necessary to compare Boogie PL to the language of regular programs. In particular, we omit procedure calls, which were also not considered within regular programs. We also omit Boogie's typing discipline. Types were anyhow ignored in the generation of verification conditions, at least in the paper [Barnett *et al.*, 2005].

We will start with the definition of what we call core Boogie programs. This Definition will be extended to Boogie programs without any qualification in Definition 104.

Definition 102 (Syntax of Core Boogie Programs) *A Core Boogie program π is a labeled sequence of commands taken from the following list:*

1. $v := t$;
2. **assume** F;
3. **havoc** v;
4. **goto** L;

where v stand for a (program) variable, t a term, F a formula in a first order language, and L for a set of labels.
Every command in π is prefixed by a unique label.

In the original definition a program is composed of blocks, i.e., sequences of commands without jumps, and sets of block labels occur as parameters of goto commands. Thus, our version corresponds to Boogie programs where every line is a block of its own.

For the purposes of the semantics of Boogie programs a state s is a mapping from program variables to values. Again the domain of values as well as the interpretation of the function and predicate symbols of the first-order

language are determined by a computation domain $\mathcal{M}$ as in Definitions 97 and 98. There is a difference though. Among the program variables there is an implicit variable called *pc* (for program counter). Implicit means that the variable does not occur in programs only in the semantics definition. Variable *pc* takes positive integers as values that are interpreted as line numbers. We assume that every line contains exactly one labeled command. For a label l we use $line(l)$ to denote the number of the line where label l occurs and $label(n)$ to denote the label in the line with number n. We start numbering lines from 0. If n is the last line number we use the constant end_π to stand for $n+1$. In principle we could have used line numbers as labels, thus avoiding the additional *line* and *label* functions but that would have made giving examples very tediuos because of the necessary shifting of line numbers.

Definition 103 (Semantics of Core Boogie Programs)
Let $\pi = l_1 : c_1, \ldots, l_n c_n$ be a Boogie program with labels l_i and commands c_i. Because of the presence of goto statements the present case is a bit more involved than Definition 98. In particular the semantics function ρ all the time depends on the whole *program π. Symbolic execution from top to bottom is not possible.*

Single steps *the pair (s_1, s_2) will be in $\rho_0(\pi)$ if s_2 is reached from s_1 by executing* one *command of π. Which command is executed depends on the value of pc in state s_1.*

Case *line $s_1(pc)$ is* $l : v := t;$

$$(s_1, s_2) \in \rho_0(\pi) \quad \textit{iff} \quad s_2(x) = \begin{cases} val_{\mathcal{M},s_1}(t) & \textit{if } x = v \\ s_1(pc) + 1 & \textit{if } x = pc \\ s_1(x) & \textit{otherwise} \end{cases}$$

Case *line $s_1(pc)$ is* $l :$ **assume** $F;$

$$(s_1, s_2) \in \rho_0(\pi) \quad \textit{iff} \quad (\mathcal{M}, s_1) \models F \textit{ and}$$
$$s_2(x) = \begin{cases} s_1(pc) + 1 & \textit{if } x = pc \\ s_1(x) & \textit{otherwise} \end{cases}$$

Case *line $s_1(pc)$ is* $l :$ **havoc** $v;$

$$(s_1, s_2) \in \rho_0(\pi) \quad \textit{iff} \quad s_2(x) = \begin{cases} \textit{arbitrary} & \textit{if } x = v \\ s_1(pc) + 1 & \textit{if } x = pc \\ s_1(x) & \textit{otherwise} \end{cases}$$

Case *line* $s_1(pc)$ *is* l : **goto** L;

$$(s_1, s_2) \in \rho_0(\pi) \quad \textit{iff} \quad s_2(x) = \begin{cases} line(l) & \textit{if } x = pc \\ \textit{for some } l \in L & \\ s_1(x) & \textit{if } x \neq pc \end{cases}$$

If $s_1(pc)$ *is not a line number occuring in* π *there is no* s_2 *with* $(s_1, s_2) \in \rho_0(\pi)$.

Composite programs *the pair* (s_1, s_2) *will be in* $\rho^n(\pi)$ *if* s_2 *is reached from* s_1 *by executing* n *commands of* π.

$$\begin{array}{lcl} (s_1, s_2) \in \rho^1(\pi) & \Leftrightarrow & (s_1, s_2) \in \rho_0(\pi) \\ (s_1, s_2) \in \rho^{n+1}(\pi) & \Leftrightarrow & \textit{there is a state } s \textit{ such that} \\ & & (s_1, s) \in \rho^n(\pi) \textit{ and } (s, s_2) \in \rho_0(\pi) \end{array}$$

$$(s_1, s_2) \in \rho(\pi) \quad \Leftrightarrow \quad \exists n((s_1, s_2) \in \rho^n(\pi) \textit{ and } s_2(pc) = end_\pi)$$

Notice

1. If in case that line $s_1(pc)$ is l : **assume** F we have $(\mathcal{M}, s_1) \not\models F$ then there is no s with $(s_1, s) \in \rho_0(\pi)$.
2. Typically we will use $(s_1, s_2) \in \rho(\pi)$ when $s_1(ps) = 0$, i.e., when execution of π starts with the first statement.

Example 31 (Branching) *The command* **if** (F) π_1 **else** π_2 *may be used as a macro in Boogie PL. A line in a program* π

$$\begin{array}{l} l : \textbf{if}(F)\ \pi_1\ \textbf{else}\ \pi_2 \\ m : \textit{next command} \end{array}$$

can be equivalently replaced by

$$\begin{array}{l} l : \textbf{goto}\ \{l_0, l_1\} \\ l_0 : \textbf{assume}\ F \\ l_{01} : \pi_1 \\ l_{02} : \textbf{goto}\ \{m\} \\ l_1 : \textbf{assume}\ \neg F \\ l_{11} : \pi_2 \\ m : \end{array}$$

All labels except l *and* m *are assumed to be new. If* l *is the label of the last line, then also* m *will be a new label for a line with a vacuous command, e.g.,* $x := x$.

Example 32 (Loops) *The command* **while**(F) π_1 *may be used as a macro in Boogie PL. A line in a program* π

$$\begin{array}{l} l : \textbf{while}(F)\ \pi_1 \\ m : \textit{next command} \end{array}$$

can be equivalently replaced by

$$\begin{array}{l} l : \textbf{goto}\ \{l_0, l_1\} \\ l_0 : \textbf{assume}\ F \\ l_{01} : \pi_1 \\ l_{02} : \textbf{goto}\ \{l\} \\ l_1 : \textbf{assume}\ \neg F \\ m : \end{array}$$

Again all labels except l *and* m *are assumed to be new.*

So far we have seen no equivalent to the dynamic logic formulas $[\pi]F$ or $\langle\pi\rangle F$. Indeed, asserting program properties is done differently in the Boogie programming language. Instead of requiring that a formula F is true after (all) execution(s) of a program in Booge PL assertions may be placed anywhere in the program code, with the intuitive requirement that they should be true whenever they are reached by program execution.

Definition 104 (Syntax of Boogie Programs) *A Boogie program is a sequence of commands taken from the list in Definition 102 and in addition*

5. **assert** F;

where F *a formula in a first order language.*

Let S as in Definition 103 be the set of all states, i.e., all functions mapping program variables, including the implicit program counter variable *pc* to values. To extend the semantics definition to programs containing **assert** statements we need an additional state denoted by $\lightning$, that is intended to signify failure of an assertion. We are thus looking at the set of state $S^+ = S \cup \{\lightning\}$.

Definition 105 (Semantics of Boogie Programs)
Let $\pi = l_1 : c_1, \ldots, l_n : c_n$ be a Boogie program with labels l_i and commands c_i.
Single steps

Case *line $s_1(pc)$ is* $l :$ **assert** F;

$$(s_1, s_2) \in \rho_0(\pi) \quad iff \quad \begin{array}{l} (\mathcal{M}, s_1) \models F \text{ and} \\ s_2(x) = \begin{cases} s_1(pc) + 1 & \text{if } x = pc \\ s_1(x) & \text{otherwise} \end{cases} \\ \text{or} \\ (\mathcal{M}, s_1) \models \neg F \text{ and } s_2 = \lightning \end{array}$$

Composite programs

No changes in the definition of ρ^n. But, observe that $(s_1, s_2) \in \rho_0(\pi)$ implies $s_1 \neq \lightning$.

$$\begin{array}{rcl} (s_1, s_2) \in \rho(\pi) & \Leftrightarrow & \exists n((s_1, s_2) \in \rho^n(\pi) \text{ and} \\ & & (s_2(pc) = end_\pi \text{ or } s_2 = \lightning)) \end{array}$$

Definition 106 (Correctness of Boogie Programs)
We say a Boogie program π is correct if for every state $s \in S$ with $s(pc) = 0$ the pair $(s, \lightning)$ is not in $\rho(\pi)$.

Note, this notion of correctness does not include termination of π.

In the next lemma we want to compare state transitions of a regular program *pi* with a corresponding Boogie prograsm π^B. The state spaces, $S : Var \to Values$ on the DL side and $S^B : Var \cup \{pc\} \to Values$ on the Boogie side, differ only in their domains. We stipulate that the variable *pc* does not occur in π, i.e. $pc \notin Var$. For $s \in S$ we write $s + val$ for the state in S^B given by

$$(s + val)(v) = \begin{cases} s(v) & \text{if } v \neq pc \\ val & \text{if } v = pc \end{cases}$$

Lemma 90 *For every regular program π there is a Core Boogie program π^c such that for all states $s_1, s_2 \in S$ with :*

$$(s_1, s_2) \in \rho^{DL}(\pi) \quad \Leftrightarrow \quad (s_1 + 0, s_2 + end_\pi) \in \rho^{Boogie}(\pi^c)$$

Proof We use superscripts to distinguish ρ^{DL} as given in Definition 98 from ρ^{Boogie} as given in Definition 103.

Looking at items 5,6,7 in Definition 96 we see that assignment and sequential composition are common to both languages. Furthermore $F?$ in regular programs is equivalent to **assume** F in Boogie PL.

$$\begin{array}{l} l : \pi_1 \cup \pi_2; \\ m : \text{next command} \end{array}$$

can be simulated by the Boogie program

l : **goto** $\{l_1, l_2\}$;
$l_1 : \pi_1$;
l_{11} : **goto** $\{m\}$;
$l_2 : \pi_2$;
m :

$$\begin{array}{l} l : \pi^*; \\ m : \text{next command} \end{array}$$

can be simulated by

l : **goto** $\{l_1, m\ \}$;
$l_1 : \pi$;
l_{12} : **goto** $\{l\}$;
m :

∎

Lemma 91 *For every regular program π let π^c be the corresponding Core Boogie program from Lemma 90. Note, that π^c contains no* **assert** *statement. Let $\pi_1 = \pi^c$;* **assert** *F; then for all states s:*

$$\begin{array}{lcl} s \models [\pi]F & \Leftrightarrow & (s+0, \lightning) \notin \rho(\pi_1) \\ s \models \langle\pi\rangle F & \Leftrightarrow & \text{there is } s_2 \in S^B \text{ with } \neq \lightning \text{ and } (s+0, s_2) \in \rho(\pi_1) \end{array}$$

Proof Straightforward

∎

4.3 Propositional Dynamic Logic

The previous section 4.2 contains an introduction to regular Dynamic Logic. This is a first-order logic allowing universal and existential quantification. In this section we will present a propositional version of Dynamic Logic.

Definition 107 (Syntax of PDL) *Let* PVar *be a set of propositional variables.*

The set $PFml$ of formula propositional Dynamic Logic is defined by the following inductive definition:

1. *atomic formulas*
   $p \in PFml$ *for any propositional variable* $p \in \text{PVar}$.
2. *closure under propositional logic operators*
   If $F_1, F_2 \in PFml$ *then also* $F_1 \vee F_2$ $F_1 \wedge F_2$, $F_1 \rightarrow F_2$, $\neg F_1$
3. *modal operators*
   $[\pi]F, \langle\pi\rangle F \in PFml$ *for* $F \in PFml$ *and* $\pi \in \Pi$.
4. *atomic programs*
   $a \in \Pi$ *for every atomic program* $a \in \text{AP}$
5. *composite programs*
   If π_1,$\pi_2 \in \Pi$ *then*
   (a) $\pi_1; \pi_2 \in \Pi$ — *sequential composition*
   (b) $\pi_1 \cup \pi_2 \in \Pi$ — *nondeterministic choice*
   (c) $\pi^* \in \Pi$ — *iteration*
6. *tests*
   $con? \in \Pi$ *for every* $con \in PFml$.

To avoid notational clutter we still use Π to denote the set of programs for propositional Dynamic Logic. If necessary we will write more specifically Π_{PDL}.

Definition 107 refers to the new concept of *atomic programs*. In a propositional logic there are no variables for objects, and consequently there can be no assignment statements in programs. But, assignments were the basis

for programs, all other constructs did produce new programs from already existing ones. Atomic programs serve as the basis for defining more complex programs in Propositional Dynamic Logic. The semantics of atomic programs will be explained in due course below.

Definition 108 (PDL Kripke Structure) *A PDL Kripke structure*

$$\mathcal{K} = (S, \models, \rho)$$

is determined by:

S	*the set of states*
$\models \subseteq (S \times \mathrm{PVar})$	*evaluation of propositional atoms in states*
$\rho : \mathrm{AP} \to \mathcal{P}(S \times S)$	*the accessibility relations for atomic programs*

Notice the differences to Definition 97: For PDL Kripke structures there is no computational domain, since this is a first-order and not a propositional concept. In Definition 97 the interpretation ρ of programs was uniquely determined by the computational domain. Here it depends on the arbitrary interpretation $\mathrm{AP} \to \mathcal{P}(S \times S)$ of atomic programs.

The following semantics definition extends

- $\models \subseteq (S \times \mathrm{PVar})$ to a relation $\models \subseteq (S \times PFml)$ and
- ρ from a function $\mathrm{AP} \to \mathcal{P}(S \times S)$ to a function $\Pi \to \mathcal{P}(S \times S)$.

Definition 109 (PDL Semantics)

$s \models p,\ p \in PVar$	*iff*	$s \models p$
$s \models F$	*iff*	F *matching one of* $F_1 \vee F_2, F_1 \wedge F_2,$ $F_1 \to F_2, \neg F_1$ *as usual.*
$s \models [\pi]F$	*iff*	$s' \models F$ *for all* s' *with* $(s, s') \in \rho(\pi)$
$s \models \langle\pi\rangle F$	*iff*	*there exists* s' *with* $(s, s') \in \rho(\pi)$ *and* $s' \models F$
$(u, u') \in \rho(a),\ a \in AP$	*iff*	$(u, u') \in \rho(a)$
$(u, u') \in \rho(\pi_1; \pi_2)$	*iff*	*there exists* $w \in S$ *with* $(u, w) \in \rho(\pi_1)$ *and* $(w, u') \in \rho(\pi_2)$
$(u, u') \in \rho(\pi_1 \cup \pi_2)$	*iff*	$(u, u') \in \rho(\pi_1)$ *or* $(u, u') \in \rho(\pi_2)$
$(u, u') \in \rho(\pi^*)$	*iff*	*there exists* n *and* $u_1, \ldots u_n \in S$ *such that* $u_1 = u$ *and* $u_n = u'$ *and* $(u_i, u_{i+1}) \in \rho(\pi)$ *for* $1 \leq i < n$
$(u, u') \in \rho(con?)$	*iff*	$u = u'$ *and* $u \models con$

Lemma 92 (Tautologies of PDL)

1. $[\pi_1;\pi_2]F \leftrightarrow [\pi_1][\pi_2]F$
2. $[\pi_1 \cup \pi_2]F \leftrightarrow ([\pi_1]F \wedge [\pi_2]F)$
3. $[(\pi)^*]F \leftrightarrow (F \wedge [\pi][(\pi)^*]F)$
4. $\langle\pi\rangle F \leftrightarrow \neg[\pi]\neg F$
5. $\langle\pi_1;\pi_2\rangle F \leftrightarrow \langle\pi_1\rangle\langle\pi_2\rangle F$
6. $\langle\pi_1 \cup \pi_2\rangle F \leftrightarrow (\langle\pi_1\rangle F \vee \langle\pi_2\rangle F)$
7. $\langle(\pi)^*\rangle F \leftrightarrow (F \vee \langle\pi\rangle\langle(\pi)^*\rangle F)$
8. $[\pi](F \to G) \to ([\pi]F \to [\pi]G)$
9. $[(\pi)^*](F \to [\pi]F) \to (F \to [(\pi)^*]F)$

Proofs The proofs are easy and straightforward. The reader may want to try finding them himself before reading on.
For all proofs we look at an arbitrary PDL Kripke structure $\mathcal{K}$ and an arbitrary state s of $\mathcal{K}$.

Proof of (1).

$$\begin{array}{rcl} s \models [\pi_1;\pi_2]F & \Leftrightarrow & \text{for all } t \text{ with } (s,t) \in \rho(\pi_1;\pi_2)\ t \models F \\ & \Leftrightarrow & \text{for all } t_1, t_2 \text{ with } (s,t_1) \in \rho(\pi_1) \text{ and } (t_1,t_2) \in \rho(\pi_2) \\ & & \quad t_2 \models F \\ & \Leftrightarrow & \text{for all } t_1 \text{ with } (s,t_1) \in \rho(\pi_1)\ t_1 \models [\pi_2]F \\ & \Leftrightarrow & s \models [\pi_1][\pi_2]F \end{array}$$

Proof of (2).

$$\begin{array}{rcl} s \models [\pi_1 \cup \pi_2]F & \Leftrightarrow & \text{for all } t \text{ with } (s,t) \in \rho(\pi_1 \cup \pi_2)\ t \models F \\ & \Leftrightarrow & \text{for all } t \text{ with } (s,t) \in \rho(\pi_1) \text{ or } (s,t) \in \rho(\pi_2)\ t \models F \\ & \Leftrightarrow & \text{for all } t \text{ with } (s,t) \in \rho(\pi_1)\ t \models F \text{ and} \\ & & \quad \text{for all } t \text{ with } (s,t) \in \rho(\pi_2)\ t \models F \\ & \Leftrightarrow & s \models [\pi_1]F \wedge [\pi_2]F \end{array}$$

Proof of (3).

$$
\begin{array}{rcl}
s \models [(\pi)^*]F & \Leftrightarrow & \text{for all } n \in \mathbb{N} \text{ and } t_1, \ldots, t_n \text{ with } s = t_1 \text{ and} \\
& & (t_i, t_{i+1}) \in \rho(\pi) \text{ for all } 1 \leq i < n \text{ we have } t_n \models F \\
& \Leftrightarrow & s \models F \text{ and} \\
& & \text{for all } n \in \mathbb{N}\ n > 0 \text{ and } t_1, \ldots, t_n \text{ with } s = t_1 \text{ and} \\
& & (t_i, t_{i+1}) \in \rho(\pi) \text{ for all } 1 \leq i < n \text{ we have } t_n \models F \\
& \Leftrightarrow & s \models F \text{ and} \\
& & \text{for all } t_1 \text{ with } (s, t_1) \in \rho(\pi) \text{ it is true that} \\
& & \text{for all } n \in \mathbb{N} \text{ and } t_2, \ldots, t_n \text{ with } t_1 = t_2 \text{ and} \\
& & (t_i, t_{i+1}) \in \rho(\pi) \text{ for all } 2 \leq i < n \text{ we have } t_n \models F \\
& \Leftrightarrow & s \models F \text{ and} \\
& & \text{for all } t_1 \text{ with } (s, t_1) \in \rho(\pi)\ t_1 \models [(\pi)^*]F \\
& \Leftrightarrow & s \models (F \wedge [\pi][(\pi)^*]F)
\end{array}
$$

Proof of (4). This is Exercise 4.7.4.

Proof of (5).

$$
\begin{array}{rcll}
\langle \pi_1; \pi_2 \rangle F & \leftrightarrow & \neg([\pi_1; \pi_2]\neg\, F) & \text{by (4)} \\
& \leftrightarrow & \neg([\pi_1][\pi_2]\neg\, F) & \text{by (1)} \\
& \leftrightarrow & \neg([\pi_1]\neg(\neg[\pi_2]\neg\, F)) & \text{logic} \\
& \leftrightarrow & \neg([\pi_1]\neg F) & \text{by (4)} \\
& \leftrightarrow & \langle \pi_1 \rangle \langle \pi_2 \rangle F & \text{again by (4)}
\end{array}
$$

Proof of (6). Use (2) and (4) in the same way as in the proof of (5).

Proof of (7). Use (3) and (4) in the same way as in the proof of (5).

Proof of (8). This is Exercise 4.7.4.

Proof of (9). Assume $s \models [(\pi)^*](F \rightarrow [\pi]F)$ and aim to show $s \models (F \rightarrow [(\pi)^*]F)$
By assumption we know for all $n \in \mathbb{N}$ and all $t_1, \ldots, t_n$ with $s = t_1$ and $(t_i, t_{i+1}) \in \rho(\pi)$ for all $1 \leq i < n$ it is true that $t_n \models (F \rightarrow [\pi]F)$. Let us keep this in mind and start the verification of $s \models (F \rightarrow [(\pi)^*]F)$. Thus we look at $s \models F$ and need to show $s \models [(\pi)^*]F)$. So we consider $m \in \mathbb{N}$ and $s_1 \ldots s_m$ with $s_1 = s$ and $(s_i, s_{i+1}) \in \rho(\pi)$ and need to show $s_m \models F$. We prove by induction that for all $1 \leq i \leq m$ $s_i \models F$ ist true. For $i = 1$ this is obvious since $s = s_1$. For the step from i to $i+1$ we start from the inductive assumption $s_i \models F$. At this point we remember our initial assumption, which yields $s_i \models (F \rightarrow [\pi]F)$. Since $s_i \models F$ is guaranteed we get $s_i \models [\pi]F$ from which we conclude $s_{i+1} \models F$.

4.4 Decidability of Propositional Dynamic Logic

Almost any propositional modal logic with just the two classical modal operators $\Box$ and $\Diamond$ is decidable. It is much harder to prove that also propositional Dynamic Logic is decidable. The guiding idea is to use again the concept of a filtration of Kripke structures. But, the details are much more involved.

Definition 110 (Normalized PDL formulas)
We call a formula $F \in PFml$ normalized *if the only propositional connectives are $\vee$ and $\neg$ and $\langle . \rangle$ is the only modal operator.*
Obviously, every F is logically equivalent to a normalized formula.

Later, the size of a PDL formula and programs will play a role in the decidability proof. Though, it is pretty obvious what the size should be, we give a precise definition to remove any ambiguities, such as: do we count parenthesis? Does $\langle\rangle$ count for one symbol or two?.

Definition 111 (Size PDL formulas)
Let PP be a normalized PDL formula or a PDL program. The size of PP, $sz(PP)$ is recursively defined as follows

$$\begin{array}{lcll} sz(p) & = & 1 & \text{for } p \in PVar \\ sz(a) & = & 1 & \text{for } a \in AP \\ sz(\neg PP) & = & sz(PP) + 1 & \\ sz(PP_1 \vee PP_2) & = & sz(PP_1) + 1 + sz(PP_2) & \\ sz(PP_1; PP_2) & = & sz(PP_1) + 1 + sz(PP_2) & \\ sz(PP_1 \cup PP_2) & = & sz(PP_1) + 1 + sz(PP_2) & \\ sz(PP^*) & = & sz(PP) + 1 & \\ sz(PP?) & = & sz(PP) + 1 & \\ sz(\langle PP_1 \rangle PP_2) & = & sz(PP_1) + 1 + sz(PP_2) & \end{array}$$

The formulas $\langle \pi_1 \rangle G$, $\langle \pi_2 \rangle G$ are not subformulas of $\langle \pi_1; \pi_1 \rangle G$, yet they have the *flavour* of subformulas. This motivates the following definition.

Definition 112 (Extended Subformulas)
Let G a normalized PDL formula. The set $ESub(G)$ of all (strict) extended subformulas *of G is inductively defined by*

$$\begin{array}{lcll}
ESub(p) & = & \{p\} & p \in PVar \\
ESub(\neg G) & = & \{\neg G\} \cup ESub(G) & \\
ESub(G_1 \vee G_2) & = & \{G_1 \vee G_2\} \cup ESub(G_1) \cup ESub(G_2) & \\
ESub(\langle a \rangle G) & = & \{\langle a \rangle G\} \cup ESub(G) & a \in AP \\
ESub(\langle \pi_1 ; \pi_2 \rangle G) & = & \{\langle \pi_1 ; \pi_2 \rangle G\} \cup ESub(\langle \pi_1 \rangle G) \cup ESub(\langle \pi_2 \rangle G) & \\
ESub(\langle \pi_1 \cup \pi_2 \rangle G) & = & \{\langle \pi_1 ; \pi_2 \rangle G\} \cup ESub(\langle \pi_1 \rangle G) \cup ESub(\langle \pi_2 \rangle G) & \\
ESub(\langle F? \rangle G) & = & \{\langle F? \rangle G\} \cup ESub(F) \cup ESub(G) & \\
ESub(\langle \pi^* \rangle G) & = & \{\langle \pi^* \rangle G\} \cup ESub(G) &
\end{array}$$

The operation ESub is extended to sets of normalized formulas by $ESub(\{F_1, \ldots, F_k\}) = ESub(F_1) \cup \ldots \cup ESub(F_k)$.

Note that we have set up this definition such that $G \in ESub)G)$.

$ESub(G)$ is a finite set of normalized PDL formulas for every normalized PDL formula G and we prove the following simple observations:

Lemma 93 *Let G be a normalized PDL formula and $F \in ESub(G)$ with $F \neq G$. Then*

$$sz(F) < sz(G)$$

Proof The proof - obviously - proceeds by structural induction on G. We use the notation $ESub_0(G) = ESub(G) \setminus \{G\}$. The claim can thus be stated as:

$$F \in ESub_0(G) \Rightarrow sz(F) < sz(G)$$

Before we start we note, that the claim implies $sz(F) \leq sz(G)$ for all $F \in ESub(G)$. This will tacitly be used in the application of the induction hypothesis.

The initial case, $G = p$, is trivially true since $ESub_0(G) = \emptyset$.
For the induction step we distinguish the remaining seven cases from Definition 112.

If $F \in ESub_0(\neg G)$ then $F \in ESub(G)$. By induction hypothesis
$sz(F) \leq sz(G) < sz(G) + 1 = sz(\neg G)$.

If $F \in ESub_0(G_1 \vee G_2)$ then $F \in ESub(G_1) \cup ESub(G_2)$.
By induction hypothesis we obtain
$sz(F) \leq sz(G_1)$ or $sz(F) \leq sz(G_2)$ and thus
$sz(F) < sz(G_1) + sz(G_2) < sz(G_1 \vee G_2)$.

If $F \in ESub_0(\langle a \rangle G)$ then $F \in ESub(G)$. By induction hypothesis
$sz(F) \leq sz(G) < sz(G) + 1 = sz(\langle a \rangle G)$.

If $F \in ESub_0(\langle \pi_1; \pi_2 \rangle G)$ then $F \in ESub(\langle \pi_1 \rangle G) \cup ESub(\langle \pi_2 \rangle G)$
By induction hypothesis
$sz(F) \leq sz(\langle \pi_1 \rangle G) = sz(\pi_1) + sz(G) + 1$ or
$sz(F) \leq sz(\langle \pi_2 \rangle G) = sz(\pi_2) + sz(G) + 1$,
and thus $sz(F) < sz(\pi_1) + sz(\pi_2) + sz(G) + 1$ and furthermore
$sz(F) < sz(\langle \pi_1; \pi_2 \rangle G) = sz(\pi_1) + sz(\pi_2) + sz(G) + 2$.

If $F \in ESub_0(\langle \pi_1 \cup \pi_2 \rangle G)$ then $F \in ESub(\langle \pi_1 \rangle G) \cup ESub(\langle \pi_2 \rangle G)$
Completely analogous to the previous case.

If $F \in ESub_0(\langle G_1? \rangle G_2)$ then $F \in ESub(G_1) \cup ESub(G_2)$.
By induction hypothesis
$sz(F) \leq sz(G_1)$ or $sz(F) \leq sz(G_2)$ and thus
$sz(F) < sz(G_1 + sz(G_2) < sz(\langle G_1? \rangle G_2) = sz(G_1 + sz(G_2) + 2$.

If $F \in ESub_0(\langle \pi^* \rangle G$ then $F \in ESub(G)$. By induction hypothesis
$sz(F) \leq sz(G) < sz(\langle \pi^* \rangle G) = sz(\pi) + sz(G) + 2$.

∎

Lemma 94 *Let F_0 be a normalized PDL formula and $F_1 \in ESub(F_0)$ and $F_2 \in ESub(F_1)$ then $F_2 \in ESub(F_0)$.*

Proof The proof proceeds by a lengthy structural induction on F_0.

$\mathbf{F_0 = p}$
Then $F_1 = F_2 = p$.

We observe that the case $F_1 = F_0$ is trivially true. In all of the following arguments we will thus concentrate on the case $F_1 \neq F_0$.

$\mathbf{F_0 = \neg F_{00}}$
Thus $F_1 \in ESub(F_{00})$ and by induction hypothesis $F_2 \in ESub(F_{00})$. Since $ESub\neg F_{00} = \{\neg F_{00}\} \cup ESub(F_{00})$ we immediately get $F_2 \in ESub(\neg F_{00})$, as desired.

$\mathbf{F_0 = F_{01} \vee F_{02}}$
By definition of $ESub(F_0)$ we obtain $F_1 \in ESub(F_{01})$ or $F_1 \in ESub(F_{02})$ and by induction hypothesis $F_2 \in ESub(F_{01})$ or $F_2 \in ESub(F_{02})$. Again by definition of $ESub(F_0)$ this gives $F_2 \in ESub(F_0)$.

$\mathbf{F_0 = \langle a \rangle F_{00}}$
By definition of $ESub(F_0$ we need to consider $F_1 \in ESub(F_{00}$, by induction hypothesis $F_2 \in ESub(F_{00}$, and again resorting to the definition of $ESub(F_0$ we get $F_2 \in ESub(F_0$.

$\mathbf{F_0 = \langle \pi_1; \pi_2 \rangle F_{00}}$
By definition of $ESub(F_0)$ we obtain $F_1 \in ESub(\langle \pi_1 \rangle F_{00})$ or $F_1 \in ESub(\langle \pi_2 \rangle F_{00})$ and by induction hypothesis $F_2 \in ESub(\langle \pi_1 \rangle F_{00})$ or $F_2 \in ESub(\langle \pi_2 \rangle F_{00})$. This immediately results in $F_2 \in ESub(\langle \pi_1; \pi_2 \rangle F_{00})$.

$\mathbf{F_0 = \langle \pi_1 \cup \pi_2 \rangle F_{00}}$
Completely analogous to the previous case.

The remaining cases $\mathbf{F_0 = \langle F_{01}? \rangle F_{00}}$ and $\mathbf{F_0 = \langle \pi^* \rangle F_{00}}$ follow the same pattern. We skip them.

■

For a more fine grained an analysis of $ESub$ we need to define subprograms of a program π. Since via tests also formulas occur in programs we even need two concepts: subprograms $SProg(\pi)$ of π and subformulas $SFml(\pi)$ of π.

Definition 113 *Let $\pi \in \Pi$ be a program.*

$$
\begin{array}{lcll}
SProg(a) & = & \{a\} & \text{if } a \in AP \\
SProg(\pi_1; \pi_2) & = & \{\pi_1; \pi_2\} \cup SProg(\pi_1) \cup SProg(\pi_2) & \\
SProg(\pi_1 \cup \pi_2) & = & \{\pi_1 \cup \pi_2\} \cup SProg(\pi_1) \cup SProg(\pi_2) & \\
SProg(\pi^*) & = & \{\pi^*\} & \\
SProg(G?) & = & \{G?\} &
\end{array}
$$

$$
\begin{array}{lcll}
SFml(a) & = & \emptyset & \text{if } a \in AP \\
SFml(\pi_1; \pi_2) & = & SFml(\pi_1) \cup SFml(\pi_2) & \\
SFml(\pi_1 \cup \pi_2) & = & SFml(\pi_1) \cup SFml(\pi_2) & \\
SFml(\pi^*) & = & \emptyset & \\
SFml(G?) & = & ESub(G) &
\end{array}
$$

Lemma 95 *Let π be a program.*

1. *For every $\pi' \in SProg(\pi)$ with $\pi \neq \pi$:* $sz(\pi') < sz(\pi)$.

2. For every $G \in SProg(\pi)$: $sz(G) < sz(\pi)$.

Proof Easy

■

Lemma 96 *For any program $\pi \in \Pi$ and formula G:*

$$ESub(\langle\pi\rangle G) = \{\langle\pi_0\rangle G \mid \pi_0 \in SProg(\pi)\} \cup SFml(\pi) \cup ESub(G)$$

Proof The proof proceeds by structural induction on π.
For $\pi = a$ we get

$$\begin{array}{lcll} SProg(\pi) & = & \{a\} & \\ \{\langle\pi_0\rangle G \mid \pi_0 \in SProg(\pi)\} & = & \{\langle a\rangle G\} & \\ SFml(\pi) & = & \emptyset & \\ ESub(\langle\pi\rangle G) & = & \{\langle a\rangle G\} \cup ESub(G) & \text{Def. of } ESub \end{array}$$

For $\pi = \pi_1; \pi_2$ we get

$$\begin{array}{lcll} ESub(\langle\pi\rangle G) & = & \{\langle\pi\rangle G\} \cup ESub(\langle\pi_1\rangle G) \cup ESub(\langle\pi_2\rangle G) & \\ & & & \text{Def. of } ESub \\ & = & \{\langle\pi'\rangle G \mid \pi' \in \{\pi\} \cup SProg(\pi_1) \cup SProg(\pi_2)\} & \\ & & \cup\ SFml(\pi_1) \cup SFml(\pi_2) \cup ESub(G) & \\ & & & \text{Ind.Hyp} \\ & = & \{\langle\pi'\rangle G \mid \pi' \in SProg(\pi_1; \pi_2)\} & \\ & & & \text{Def. of } SProg \\ & & \cup\ SFml(\pi_1) \cup SFml(\pi_2) \cup ESub(G) & \\ & = & \{\langle\pi'\rangle G \mid \pi' \in SProg(\pi)\} \cup SFml(\pi) \cup ESub(G) & \\ & & & \text{Def. of } SFml \end{array}$$

The case $\pi = \pi_1 \cup \pi_2$ is completely analogous to the previous one.

For $\pi = \pi_1^*$ we get

$$\begin{array}{lcll} ESub(\langle\pi\rangle G) & = & \{\langle\pi\rangle G\} \cup ESub(G) & \text{Def. of } ESub \\ & = & \{\langle\pi'\rangle G \mid \pi' \in \{\pi\}\} \cup \emptyset \cup ESub(G) & \text{simple} \\ & = & \{\langle\pi'\rangle G \mid \pi' \in SProg\pi)\} \cup & \text{Def. of } SProg \\ & & SFml(\pi) \cup ESub(G) & \text{Def. of } SFml \end{array}$$

For $\pi = G_1?$ we get

$$\begin{array}{rcll} ESub(\langle\pi\rangle G_2) & = & \{\langle\pi\rangle G_2\} \cup ESub(G_1) \cup ESub(G_2) & \text{Def. of } ESub \\ & = & \{\langle\pi'\rangle G_2 \mid \pi' \in SProg(\pi)\} \cup & \text{Def. of } SProg \\ & & ESub(G_1) \cup ESub(G_2) & \\ & = & \{\langle\pi'\rangle G_2 \mid \pi \in SProg(\pi)\} \cup & \\ & & SFml(\pi) \cup ESub(G_2) & \text{Def.of } SProg \end{array}$$

■

One deficiency of $ESub$ is that it does not *look inside* the iteration program π^*. This limits the usefulness of the concept of extended subformulas for more advanced proofs by structural induction.

Definition 114 *Let G be a formula. We inductively define the sets $FL_n(G)$ of formulas.*

$FL_0(G) = ESub(G)$

$\Pi_n(G) = \{\pi \mid \langle\pi^*\rangle F \in FL_n(G) \setminus \bigcup_{0\leq i<n} FL_i(G)\}$

$FL'_n(G) = \{\langle\pi\rangle\langle\pi^*\rangle F \mid \langle\pi^*\rangle F \in FL_n(G) \setminus \bigcup_{0\leq i<n} FL_i(G)\}$

$FL_{n+1}(G) = ESub(FL'_n)$

$FL(G) = \bigcup_{n\geq 0} FL_n(G)$

Lemma 97 *For every normalized PDL formula there is k such that*

$$FL(G) = \bigcup_{0\leq n\leq k} FL_n(G)$$

$FL(G)$ is called the Fischer-Ladner Closure of G.

Proof Let us look at $\pi \in \Pi_{n+1}(G)$. By Definition 114 there is a formula F such that $\langle\pi^*\rangle F \in FL_{n+1}(G)$ and $\langle\pi^*\rangle F \notin FL_i(G)$ for all $i \leq n$. Since $FL_{n+1}(G) = ESub(FL'_n(G))$ we get from the definition of $FL'_n(G)$ a formula F_0 and a program π_0 such that $\langle\pi_0^*\rangle F_0 \in FL_n(G) \setminus \bigcup_{0\leq i<n} FL_i(G)$ and $\langle\pi^*\rangle F \in ESub(\langle\pi_0\rangle\langle\pi_0^*\rangle F_0)$. We note for later reference that this also entails $\pi_0 \in \Pi_n$. By Lemma 96 there thus three cases

1. $\langle\pi^*\rangle F \in \{\langle\pi_1\rangle\langle\pi_0^*\rangle F_0 \mid \pi_1 \in SProg(\pi_0)\}$
   Thus $\pi^* \in SProg(\pi_0)$ and since $\pi^* \neq \pi_0$ (otherwise $\pi_0^* \in SProg(\pi_0)$) by Lemma 95 $sz(\pi) < sz(\pi^*) < sz(\pi_0)$.

2. $\langle\pi^*\rangle F \in SFml(\pi_0)$
   We obtain directly from Lemma 95 $sz(\pi^*) < sz(\langle\pi^*\rangle F) < sz(\pi_0)$.

3. $\langle\pi^*\rangle F \in ESub(\langle\pi_0^*\rangle F_0)$
   From $\langle\pi_0^*\rangle F_0 \in FL_n(G) = ESub(FL'_{n-1}(G))$ and transitivity this would entail $\langle\pi^*\rangle F \in FL_n(G)$ contradicting the choice of F. So, this case cannot arise.

In total we have shown that for any $\pi_m \in \Pi_m(G)$ there are $\pi_i \in \Pi_i(G)$ such that $sz(\pi_0) > \ldots > sz(\pi_i) > \ldots > sz(\pi_m)$. In other words for $m > max\{sz(\pi) \mid \pi \in \Pi_0(G)\}$ we must have $\Pi_m(G) = \emptyset$, which also entails $FL'_m(G) = FL_{m+1}(G) = \emptyset$.

∎

Lemma 98 (Fischer-Ladner Closure)
Let S_0 be a set of normalized formulas in $PFml$.
The Fischer-Ladner closure of S_0 is the smallest subset $S \subseteq PFml$ satisfying:

1	$S_0 \subseteq S$		
2	$\neg G \in S$	$\Rightarrow$	$G \in S$
3	$(G_1 \vee G_2) \in S$	$\Rightarrow$	$G_1 \in S$ *and* $G_2 \in S$
4	$\langle\pi\rangle G \in S$	$\Rightarrow$	$G \in S$
5	$\langle\pi_1;\pi_2\rangle G \in S$	$\Rightarrow$	$\langle\pi_1\rangle\langle\pi_2\rangle G \in S$
6	$\langle\pi_1 \cup \pi_2\rangle G \in S$	$\Rightarrow$	$\langle\pi_1\rangle G \in S$ *and* $\langle\pi_2\rangle G \in S$
7	$\langle\pi_1^*\rangle G \in S$	$\Rightarrow$	$\langle\pi_1\rangle\langle\pi_1^*\rangle G \in S$
8	$\langle G_1?\rangle G_2 \in S$	$\Rightarrow$	$G_1 \in S$ *and* $G_2 \in S$

For normalized $F \in PFml$ we denote by $\mathrm{FL}(F)$ *the Fischer-Ladner closure of $\{F\}$.*

Example 33

We compute the Fischer-Ladner closure for $G = (p \to \langle (q?;a)^*;\neg q?\rangle r$

$$\begin{array}{rcl}
FL_0(G) & = & FL_0(\neg p \vee \langle (q?;a)^*;\neg q?\rangle r) \\
& = & ESub(\neg p \vee \langle (q?;a)^*;\neg q?\rangle r) \\
& = & \{G\} \cup ESub(\neg p) \cup ESub(\langle (q?;a)^*;\neg q?\rangle r) \\
& = & \{G, \neg p, p\} \cup ESub(\langle (q?;a)^*;\neg q?\rangle r) \\
& = & \{G, \langle (q?;a)^*;\neg q?\rangle r, \neg p, p\} \cup \\
& & ESub(\langle (q?;a)^*\rangle r) \cup ESub(\langle \neg q?\rangle r) \\
& = & \{G, \langle (q?;a)^*\rangle r,\ \langle (q?;a)^*;\neg q?\rangle r,\ \neg p,\ p\} \cup \\
& & ESub(r) \cup ESub(\langle \neg q?\rangle r) \\
& = & \{G, \langle (q?;a)^*\rangle r,\ \langle (q?;a)^*;\neg q?\rangle r,\ \neg p,\ p\} \cup \\
& & \{\langle \neg q?\rangle r\} \cup ESub(r) \cup ESub(\neg q?) \\
& = & \{G, \langle (q?;a)^*\rangle r,\ \langle (q?;a)^*;\neg q?\rangle r,\ \langle \neg q?\rangle r\} \cup \\
& & \{\ \neg p,\ p,\ \neg q,\ q,\ r\} \\
\Pi_0(G) & = & \{q?;a\} \\
FL'_0(G) & = & \{\langle q?;a\rangle \langle (q?;a)^*\rangle r\} \\
\multicolumn{3}{l}{\textit{We omit from here on formulas already in} FL_0(G)} \\
FL_1(G) & = & ESub(FL'_0(G)) \\
& = & ESub(\langle q?;a\rangle \langle (q?;a)^*\rangle r) \\
& = & ESub(\langle q?\rangle \langle (q?;a)^*\rangle r) \cup ESub(\langle a\rangle \langle (q?;a)^*\rangle r) \\
& = & \{\langle q?\rangle \langle (q?;a)^*\rangle r,\ \langle a\rangle \langle (q?;a)^*\rangle r\} \cup \\
& & ESub(q) \cup ESub(\langle (q?;a)^*\rangle r) \cup ESub(r) \\
& = & \{\langle q?\rangle \langle (q?;a)^*\rangle r,\ \langle a\rangle \langle (q?;a)^*\rangle r\} \\
\Pi_1(G) & = & \emptyset
\end{array}$$

In total:

$$\begin{array}{rcl}
FL(G) & = & \{G, \langle (q?;a)^*\rangle r,\ \langle (q?;a)^*;\neg q?\rangle r,\ \langle \neg q?\rangle r\} \cup \\
& & \{\langle q?\rangle \langle (q?;a)^*\rangle r,\ \langle a\rangle \langle (q?;a)^*\rangle r\} \cup \\
& & \{\ \neg p,\ p,\ \neg q,\ q,\ r\}
\end{array}$$

Definition 115 (Equivalent States)
Let $\mathcal{K} = (S, \models, \rho)$ *be a PDL Kripke structure, and* Γ *a subset of* $PFml$. *The relation* $\sim_\Gamma$ *on* S *is defined by:*

$$s_1 \sim_\Gamma s_2 \textit{ iff } s_1 \models F \Leftrightarrow s_2 \models F \textit{ for all } F \in \Gamma$$

Lemma 99
For every Γ *the relation* $\sim_\Gamma$ *is an equivalence relation on* S.

Proof The claim easily follows from the fact the $\Leftrightarrow$ is an equivalence relation.

∎

Definition 116 (Filtration)
Let Γ be set of PDL formulas such that for all atomic programs a and all F $\langle a\rangle F \in \Gamma$ implies $F \in \Gamma$. The filtration $\mathcal{K}_\Gamma = (S_\Gamma, \models_\Gamma, \rho_\Gamma)$ of $\mathcal{K} = (S, \models, \rho)$ with respect Γ is defined by:

$[s]$	$=$	$\{s' \mid s \sim_\Gamma s'\}$	*equiv. class of* s
S_Γ	$=$	$\{[s] \mid s \in S\}$	
$[s] \models_\Gamma p$	$\Leftrightarrow$	$s \models p$	*for* $p \in \Gamma$
$[s] \models_\Gamma p$		*false (arbitrary)*	*otherwise*
$([s_1], [s_2]) \in \rho_\Gamma(a)$	*iff*	*for all* $\langle a\rangle F \in \Gamma$	$a \in AP$
		if $s_1 \models \neg\langle a\rangle F$ *then* $s_2 \models \neg F$	

In the end we will use filtrations only for sets Γ that are Fischer-Ladner closures. But, this definition at least works for more general Γ.

Lemma 100
Definition 116 is independent of the choice of representatives.

Proof Let s_1, s_2, s_1', s_2' be states in S with $s_i \sim_\Gamma s_i'$ for $i \in \{1, 2\}$. We need to convince ourselves that

1. $s_1 \models p \Leftrightarrow s_1' \models p$
2. for all $\langle a\rangle F \in \Gamma$
   if $s_1 \models \neg\langle a\rangle F$ implies $s_2 \models \neg F$
   then also $s_1' \models \neg\langle a\rangle F$ implies $s_2' \models \neg F$

If $p \notin \Gamma$ we chose $s \not\models p$ for all $s \in S$. If $p \in \Gamma$, then we have by definition of equivalent states $s_i \models p \Leftrightarrow s_i' \models p$. This settles (1). To prove (2) consider $\langle a\rangle F \in \Gamma$ and assume $s_1 \models \neg\langle a\rangle F$ implies $s_2 \models \neg F$ and $s_1' \models \neg\langle a\rangle F$. We need to show $s_2' \models \neg F$. From the definition of equivalent states we first obtain $s_1 \models \neg\langle a\rangle F$. By the assumed implication this yields $s_2 \models \neg F$. By assumptions of the lemma we also have $F \in \Gamma$. Thus the equivalence of state s_2 and s_2' yields $s_2' \models \neg F$ as desired.

∎

Lemma 101 (Filtration Lemma)
Let
F *be PFml formula,*
$\Gamma = FL(F)$ *the Fischer-Ladner closure of* F
$\mathcal{K} = (S, \models, \rho)$ *a PDL Kripke structure*
$\mathcal{K}_\Gamma = (S_\Gamma, \models_\Gamma, \rho_\Gamma)$ *its quotient modulo* $\sim_\Gamma$,
then the following is true for all $G \in \Gamma$, $\pi \in \Pi$ *and* $s_1, s_2 \in S$

1. *The relation* $\sim_\Gamma$ *can have at most* $2^{card(\Gamma)}$ *equivalence classes. Since* Γ *is finite,* S_Γ *is also finite.*

2. $([s_1], [s_2]) \in \rho_\Gamma(\pi)$ *implies* *for all* $\langle\pi\rangle B \in \Gamma$
$s_1 \models \neg\langle\pi\rangle B \Rightarrow s_2 \models \neg B$

3. $(s_1, s_2) \in \rho(\pi)$ *entails* $([s_1], [s_2]) \in \rho_\Gamma(\pi)$

4. $s \models G$ *iff* $[s] \models G$

Proofs
Proof of (1). Obvious.

The following three claims will be proved by simultaneous induction on the complexity of α respectively G.

Proof of (2). If π is atomic the claim is reduced to Definition 116. For the induction step we need to consider four cases. In each case we assume that claims (2) and (3) are true for programs α and β and that (4) is true for formula F.
Case $\pi \equiv \alpha; \beta$
Assume $([w_1], [w_2]) \in \rho_\Gamma(\alpha; \beta)$. By definition of sequential composition there is a state u such that

$$([w_1], [u]) \in \rho_\Gamma(\alpha) \text{ and } ([u], [w_2]) \in \rho_\Gamma(\beta)$$

By induction hypothesis we know for all $\langle\alpha\rangle B \in \Gamma$ and all $\langle\beta\rangle C \in \Gamma$

$$w_1 \models \neg\langle\alpha\rangle B \Rightarrow u \models \neg B \quad (1)$$
$$u \models \neg\langle\beta\rangle C \Rightarrow w_2 \models \neg C \quad (2)$$

We need to show for all $\langle\alpha; \beta\rangle F \in \Gamma$ that $w_1 \models \neg\langle\alpha; \beta\rangle F$ implies $w_2 \models \neg F$.

So let us assume $w_1 \models \neg\langle\alpha;\beta\rangle F$. By definition of the Fischer-Ladner closure $\langle\alpha\rangle\langle\beta\rangle F \in \Gamma$ and also $\langle\beta\rangle F \in \Gamma$ hold. The assumption immediately yields

$$w_1 \models \neg\langle\alpha\rangle\langle\beta\rangle F$$

From (1) with $B = \langle\beta\rangle F$ we obtain:

$$u \models \neg\langle\beta\rangle F$$

Now using (2) with $C \equiv F$ we arrive at $w_2 \models \neg F$, as desired.

Case $\pi \equiv \alpha^*$
We start with a pair $([w_1],[w_2])$ in $\rho_\Gamma(\alpha^*)$. By definition of $\rho_\Gamma(\alpha^*)$ there are states $u_0,\ldots,u_k$ such that $[w_1] = [u_0]$, $[w_2] = [u_k]$ and for all $0 \leq i < k$ we know $([u_i],[u_{i+1}]) \in \rho_\Gamma(\alpha)$. By induction hypothesis for part (2) we have for all $\langle\alpha\rangle C \in \Gamma$

$$u_i \models \neg\langle\alpha\rangle C \Rightarrow u_{i+1} \models \neg C$$

We need to show for all formulas $\langle\alpha^*\rangle B \in \Gamma$

$$w_1 \models \neg\langle\alpha^*\rangle B \Rightarrow w_2 \models \neg B \qquad (3)$$

As a start we note that the formula

$$\langle\alpha^*\rangle B \leftrightarrow B \vee \langle\alpha\rangle\langle\alpha^*\rangle B$$

is a tautology of PDL. Thus the formula we get by negating both sides of the equivalence is also a tautology;

$$\neg\langle\alpha^*\rangle B \leftrightarrow \neg B \wedge \neg\langle\alpha\rangle\langle\alpha^*\rangle B$$

To prove (3) we assume $u_0 \models \neg\langle\alpha^*\rangle B$ and will show by induction on i for $0 \leq i \leq k$ that $u_i \models \neg\langle\alpha^*\rangle B$ is true. As we have just remarked $u_i \models \neg\langle\alpha^*\rangle B$ implies also $u_i \models \neg\langle\alpha\rangle\langle\alpha^*\rangle B$. By the properties of the Fischer-Ladner closure $\langle\alpha\rangle\langle\alpha^*\rangle B$ is in Γ. Thus the induction hypothesis is applicable with $C = \langle\alpha^*\rangle B$ and yields $u_{i+1} \models \neg\langle\alpha^*\rangle B$. From $u_k \models \neg\langle\alpha^*\rangle B$ we get using the tautology shwon above $u_k \models \neg B$. Since $[u_k] = [w_2]$ this also yields $w_2 \models \neg B$.

Case $\pi \equiv \mathbf{F}?$
From $([w_1],[w_2]) \in \rho_\Gamma(\langle F?\rangle)$ we get by the semantics of the test operator $[w_1] = [w_2]$ and $[w_1] \models F$. Pick an arbitrary $\langle F?\rangle B \in \Gamma$. We assume

$w_1 \models \neg\langle F?\rangle B$ and want to show $w_2 \models \neg B$. From $w_1 \models \neg\langle F?\rangle B$ we get at least $w_1 \models \neg F \vee \neg B$. Since $w_1 \sim_\Gamma w_2$ and $F, B \in \Gamma$ we also have $w_2 \models \neg F \vee \neg B$. From $[w_1] = [w_2] \models F$ we get by part 3 $w_2 \models F$, from which $w_2 \models \neg B$ follows, as desired.

Case $\pi \equiv \alpha \cup \beta$
This is the easiest case and left to the reader.

Proof of (3). This is not a crucial step. We formulated it as an extra claim since it is of some interest in itself and it will also make the argument for part 4 a bit shorter in one case.

Assume $(w_1, w_2) \in \rho(\alpha)$.To prove $([w_1], [w_2]) \in \rho_\Gamma(\alpha)$ according to the definition of ρ_Γ we consider an arbitrary formula $\neg\langle\alpha\rangle B$ in Γ with $w_1 \models \neg\langle\alpha\rangle B$. This is equivalent to $w_1 \models [\alpha]\neg B$. Thus we get $w_2 \models \neg B$ and are finished.
Proof of (4). If $G \in \Gamma$ is a propositional variable the claim is part of the definition of $\mathcal{K}_\Gamma$. The cases of the inductive step concerning propositional connectives are trivial. We will present here the inductive step from B to $G = \langle\alpha\rangle B$.

If $w_1 \models \langle\alpha\rangle B$ holds then there is a state w_2 satisfying both $(w_1, w_2) \in \rho(\alpha)$ und $w_2 \models B$. By inductive hypothesis from part 4 we get $[w_2] \models B$ and from part 3 also $([w_1], [w_2]) \in \rho_\Gamma(\alpha)$. Thus $[w_1] \models \langle\alpha\rangle B$. To prove the reverse implication assume $[w_1] \models \langle\alpha\rangle B$. This implies the existence of a state $[w_2]$ satisfying $([w_1], [w_2]) \in \rho_\Gamma(\alpha)$ and $[w_2] \models B$. The inductive hypothesis immediately gives us $w_2 \models B$. If $w_1 \models \neg\langle\alpha\rangle B$ were true the inductive hypothesis of part 2 would yield the contradiction $w_2 \models \neg B$. Thus we must indeed have $w_1 \models \langle\alpha\rangle B$.

∎

Corollary 102 (Finite Model Property)
Let F be a PFml formula.
If F has a model, then it also has a finite model.
More precisely: If F has a model, then it also has a model with at most 2^{n_F} elements where n_F is the number of symbols in F.

Proofs Immediate consequence of Lemma 101.

∎

Theorem 103 *(Decidability of PDL)*
The problem to determine wether F is a tautology for PFml formulas F is decidable.

Proofs For given F generate all models for the signature of F with at most 2^{n_F+1} elements. If none of them satisfies $\neg F$ then $\neg F$ is by Corrolary 102 not satisfiable, i.e. $\vdash F$.

∎

The decision procedure given in the proof of Theorem 103 is of course very inefficent. A more direct decision procedure is proposed in [Abate *et al.*, 2007].

Analogeous to Definition 99 we can define logical inference for propositional dynamic logic. In the propositional case there is no distinction between an interpreted and an uninterpreted case.

Definition 117 (Logical Inference)

$G \vdash F$	*G (locally) entails F*	*for all Kripke structures $\mathcal{K} = (S, \models, \rho)$ and all $s \in S$ if $s \models G$ then also $s \models F$*
$G \vdash^g F$	*G globally entails F*	*for all $\mathcal{K} = (S, \models, \rho)$ if $s \models G$ for all $s \in S$ then $s \models F$ for all $s \in S$*

Corollary 104
The problem to determine wether $G \vdash F$ is true for PFml formulas F, G is decidable.

Proofs Immediate consequence of Theorem 103 and Exercise 4.7.6.

4.5 Alternatives in PDL

We have so far presented the standard material on PDL. In this subsection we take a moment to reflect on alternative variations of PDL.

Alternative Accessibility

The semantics of PDL was introduced in two steps. The first step was taken by Definition 108 in defining the concept of a PDL Kripke structure $\mathcal{K} = (S, \models, \rho)$. This did fix $s \models p$ for all $s \in S$ and all atoms p and $\rho(a)$ for all atomic programs a. In the second step these relations were extented to $s \models F$ for arbitrary formulas F and $\rho(\pi)$ for arbitrary programs in Definition 109. This extension was deterministic, once a PDL Kripke structure $\mathcal{K}$ was fixed the relations $s \models F$ for arbitrary formulas F and $\rho(\pi)$ for arbitrary programs was also fixed. For the nonstandard semantics, that we will introduce now, this will be different. Both definitions need to be merged into one.

Definition 118 (Nonstandard Kripke Structure)
A nonstandard PDL Kripke structure

$$\mathcal{K} = (S, \models, \rho)$$

is determined by:

S	*the set of states*
$\models \subseteq (S \times PFml)$	*evaluation of propositional formulas in states*
$\rho : \Pi \to S \times S$	*the accessibility relations for programs*

such that:

$s \models p,\ p \in PVar$	*iff*	$s(p) = \mathbf{true}$
$s \models F$	*iff*	F *matching one of* $F_1 \vee F_2, F_1 \wedge F_2,$ $F_1 \to F_2, \neg F_1$ *as usual.*
$s \models [\pi]F$	*iff*	*for all* s' *with* $(s, s') \in \rho(\pi) s' \models F$
$s \models \langle\pi\rangle F$	*iff*	*there exists* s' *with* $(s, s') \in \rho(\pi)$ *and* $s' \models F$
$(u, u') \in \rho(a),\ a \in AP$	*iff*	$(u, u') \in \rho(a)$
$(u, u') \in \rho(\pi_1; \pi_2)$	*iff*	*there exists* $w \in S$ *with* $(u, w) \in \rho(\pi_1)$ *and* $(w, u') \in \rho(\pi_2)$
$(u, u') \in \rho(\pi_1 \cup \pi_2)$	*iff*	$(u, u') \in \rho(\pi_1)$ *or* $(u, u') \in \rho(\pi_2)$
$\rho(\pi^*)$ *is reflexive and transitive relation with* $\rho(\pi) \subseteq \rho(\pi^*)$ *and* $s \models [a^*]B \leftrightarrow (B \wedge [a; a^*]B)$ $s \models [a^*]B \leftrightarrow (B \wedge [a^*](B \to [a]B))$		
$(u, u') \in \rho(con?)$	*iff*	$u = u'$ *and* $u \models$ *con*

Note, that the only difference to Definition 109 is in the treatment of $\rho(\pi^*)$. The relation $\rho(\pi^*)$ is not computed from $\rho(\pi)$, but characterized by properties.

What are the differences between the standard Kripke semantics and the nonstandard semantics?

Lemma 105 (Difference between Standard and Nonstandard Semantics)

Consider the following set of modal formulas

$$\Gamma = \{\langle a^* \rangle \neg p\} \cup \{[a^n]p \mid n \geq 0\}$$

where $a \in \mathrm{AP}$, $p \in \mathrm{PVar}$ *and* $a^n = \underbrace{a; \ldots; a}_{n \text{ times}}$.

Γ *has a nonstandard Kripke model, but* Γ *is inconsistent in standard Kripke semantics.*

Proof That Γ is inconsistent in standard Kripke semantics is obvious. The idea for the rest of the proof is as follows: There is a set of first-order axioms Σ_{ns} such that there is a one-to-one correspondence between models of Σ_{ns} and nonstandard Kripke structures. The same translation can be used to find the first-order equivalent Γ' of Γ. Since every finite subset of $\Gamma' \cup \Sigma_{ns}$ is consistent the set itself is consistent by the compactness property of first-order logic. A model of $\Gamma' \cup \Sigma_{ns}$ yields the required nonstandard Kripke model of Γ.

∎

Alternative set of Possible Worlds

Frequently, the set of possible worlds in a concrete application context is uniquely determined by the set of propositional atoms and finite since only finitely many atoms are relevant. Altogether this leads to Kripke structures with a finite set of possible word. In this paragraph we assume that the set PVar is finite.

Definition 119 (State Vector Kripke Structure)

1. *A PDL Kripke structure* $\mathcal{K} = (S, \rho, \models)$ *is called a* state vector *Kripke structure if* $S \subseteq 2^{\mathrm{PVar}}$.

2. *A formula* F *is state vector satisfiable if there is a state vector Kripke structure* $\mathcal{K} = (S \subseteq 2^{\mathrm{PVar}}, \rho, \models)$ *and* $s \in S$ *with* $s \models F$.

3. F *is a state vector consequence of a set of formulas* H*, in symbols* $H \vdash_{sv} F$*, if for any state vector Kripke structure* $\mathcal{K} = (S \subseteq 2^{\mathrm{PVar}}, \rho, \models)$ *and* $s \in S$ *with* $(\mathcal{K}, s) \models H$ *we also have* $(\mathcal{K}, s) \models F$.

Obviously, a PDL tautology is also a tautology for state vector Kripke structure. But, is the converse also true? The answer is *no*.

Lemma 106
Let $U \subseteq \mathrm{PVar}$ *be a subset of the set of propositional atoms.*
Let $state_U$ *abbreviate the formula* $\bigwedge_{p \in U} p \wedge \bigwedge_{p \notin U} \neg p$
and F *an arbitrary PDL formula,* π *an arbitrary program. Then*

$$\langle \pi \rangle (state_U \wedge F) \rightarrow [\pi](state_U \rightarrow F)$$

is true in all state vector Kripke structure.

Proof Let $\mathcal{K} = (S \subseteq 2^{\mathrm{PVar}}, \rho, \models)$ be a state vector Kripke structure and $s_0 \in S$ such that $s_0 \models \langle \pi \rangle (state_U \wedge F)$. This implies the existence of $s \in S$ with $(s_0, s) \in \rho(\pi)$ and $s \models state_u \wedge F$. We need to show $s_0 \models [\pi](state_U \rightarrow F$. So consider $s' \in S$ with $(s_0, s') \in \rho(\pi)$ and $s' \models state_U$. We need to show $s' \models F$. But, $s \models state_U$ and $s' \models state_U$ imply that $s = s'$. In total this shows $s \models \langle \pi \rangle (state_U \wedge F) \rightarrow [\pi](state_U \rightarrow F)$.

∎

We will next show that the formulas refered to in Lemma 106 are in a sense the only difference between the state vector and the unrestricted semantics.

Theorem 107 *(State Vector Semantics)*
Let $\mathrm{AP} = \{a_1, \ldots, a_k\}$ *and let* π_{all} *stand for the program* $(a_1 \cup \ldots \cup a_k)^*$. *Let* H *be the set of all formulas*

$$\langle \pi_{all} \rangle (state_U \wedge F) \rightarrow [\pi_{all}](state_U \rightarrow F)$$

for all $U \subseteq \mathrm{PVar}$ *and the notation from Lemma 106. Finally, consider an arbitrary PDL formula* F.

A. $\{F\} \cup H$ *is satisfiable iff* F *is state vector satisfiable.*

B. $H \vdash F$ *iff* $\vdash_{sv} F$.

Proofs

ad A The implication $\rightarrow$ is obvious.

Let $\mathcal{K} = (S, \rho, \models)$ be a Kripke structure, $s \in S$ with $(\mathcal{K}, s) \models F$. We will produce a statisfying state vector Kripke structure for F. In a first step we consider the set S_0 of states that are recursively reachable from s via atomic programs. Formally:

$$\begin{array}{lcl} S_0^0 & = & \{s\} \\ S_0^{i+1} & = & \{s' \in S \mid \text{there are } a \in \mathrm{AP} \text{ and } s_0 \in S_0^i \text{ with } (s_0, s') \in \rho(a)\} \cup S_0^i \\ S_0 & = & \bigcup_{i \geq 0} S_0^i \end{array}$$

Let $\mathcal{K}_0 = (S_0, \rho_0, \models_0)$ be the restriction of $\mathcal{K}$ to the set of world S_0. It is easy to see that

1. $(s_1, s_2) \in \rho(\pi)$ and $s_1 \in S_0$ implies $s_2 \in S_0$ for all programs π. (This fact makes the construction of $\mathcal{K}_0$ possible in the first place.)
2. for all formulas F and all states $s' \in S_0$ $(\mathcal{K}_0, s') \models F$ iff $(\mathcal{K}, s') \models F$

From now on we will work with $\mathcal{K}_0$. No rigor is lost if we use in the following ρ for ρ_0 and $\models$ for $\models_0$. We have $(K_0, s) \models F$ to start with and we know by construction of S_0 that for any $s' \in S_0$ that $(s, s') \in \rho(\pi_{all})$.

We follow the general idea of the Filtrationlemma (Lemma 101), though some of the details will crucially differ. Two states $s_1, s_2 \in S_0$ are congruent, in symbols $s_1 \sim_{sv} s_2$, if for all $p \in \mathrm{PVar}$ $s_1 \models p$ iff $s_2 \models p$. By $[s']$ we denote the congruence class $\{s'' \mid s'' \sim_{sv} s'\}$ and we define the structure $\mathcal{K}_{quot} = (S_{quot}, \rho_{quot}, \models_{quot})$ mimicking Definition 116 by

$$\begin{array}{lcll} S_{quot} & = & \{[s'] \mid s' \in S_0\} & \\ [s'] \models_{quot} p & \Leftrightarrow & s' \models p & \text{for } p \in \mathrm{PVar} \\ ([s_1], [s_2]) \in \rho_{quot}(a) & \text{iff} & \text{for all formulas } F \text{ and} & a \in AP \\ & & \text{if } s_1 \models [a]F \text{ then } s_2 \models F & \end{array}$$

As in Lemma 101 we will prove:

1. $S_{quot} \subseteq 2^{\mathrm{AP}}$

2. $([s_1],[s_2]) \in \rho_{quot}(\pi)$ implies for all formulas F and all programs π
   $s_1 \models [\pi]F \Rightarrow s_2 \models F$

3. $(s_1, s_2) \in \rho(\pi)$ entails $([s_1],[s_2]) \in \rho_{quot}(\pi)$

4. for all $s' \in S_0$ and all formulas F
   $s' \models F$ iff $[s'] \models_{qot} F$

Surprisingly, all the proofs go through as before even though this time there is no given set of formulas Γ and we need not to make use of the assumptions H. The crucial point is that we have not yet convinced ourselves that the above definition of ρ_{quot} is independent of representatives. So consider $s_1, s_2, s_1', s_2' \in S_0$ with $s_1 \sim_{sv} s_1'$ and $s_2 \sim_{sv} s_2'$ such that for all F $s_1 \models [a]F$ implies $s_2 \models F$ for some $a \in \mathrm{AP}$. Assume $s_1' \models [a]F$ with the goal of showing $s_2' \models F$.

Let $U_i = \{p \in \mathrm{AP} \mid s_i' \models p\}$ for $i = 1, 2$. Thus $s_i' \models state_{U_i}$ and because of $s_i \sim_{sv} s_i'$ also $s_i \models state_{U_i}$. By definition of S_0 we have $(s, s_1') \in \rho(\pi_{all})$. Thus $s \models \langle \pi_{all} \rangle (state_{U_1} \wedge [a]F)$. Since $s \models H$ by assumption we also get $s \models [\pi_{all}](state_{U_1} \rightarrow [a]F)$. Because, again by definition of S_0, also $(s, s_1) \in \rho(\pi_{all})$ is true we obtain $s_1 \models (state_{U_1} \rightarrow [a]F)$. As we have shown above $s_1 \models state_{U_1}$ this yields $s_1 \models [a]F$. By the initial assumption of this proof of independence from the representatives we obtain $s_s \models F$. By the same argument using U_2 instead of U_1 we conclude $s_s' \models F$.

ad B Since H is true in any state vector Kripke structure (Lemma 106) the implication from left to right is obvious. For the reverse implication assume $\vdash_{sv} F$. If $H \vdash F$ is not true then $H \cup \{\neg F\}$ is satisfiable. By part (1) this implies that contradiction that $\neg F$ should be state vector satisfiable.

∎

4.6 Axiomatizations of Dynamic Logic

Failure of the Compactness Theorem

In this chapter we are back to full first-order dynamic logic.

Let us start by analysing the following infinite set of DL formulas

$$\{\langle \textbf{while } p(x) \textbf{ do } x := f(x)\rangle \textbf{true}\} \cup \{p(f^n(x)) \mid n \geq 0\}$$

Let $\mathcal{K}_{\mathcal{M}}$ by a Kripke structure and s one of its states such that all formulas in this set are true in state s. We denote the value of the program variable x in state s by $a = s(x)$. Thus in the domain $\mathcal{M}$ of computation we must have $p^{\mathcal{M}}(a_n)$ for all n and $a_n = (f^{\mathcal{M}})^n(a)$. The while-loop starts with a the value of x. After n iterations of the loop body x holds the value a_n. The first formula in the set says that the while-loop terminates, i.e., for some n the loop condition $p^{\mathcal{M}}(a_n)$ should be false. A contradiction. The considered set of formulas has no model, it is inconsistent. On the other hand it is easily seen that every finite subset has a model. We have thus proved:

Lemma 108

1. *The logic* $\mathbf{DL}_{reg}$ *does not satisfy the compactness theorem, i.e., there is an inconsistent set of formulas with all its finite subsets being consistent.*

2. *There is no sound and complete calculus for* $\mathbf{DL}_{reg}$.

Proof It remains to consider the second claim. Assume there is a complete and correct calculus for $\mathbf{DL}_{reg}$ and S is an inconsistent infinite set of $PModFml$-formulas. By completeness there is a derivation of the contradiction $p \wedge \neg p$ from the assumptions S. Since any derivation, no matter how it is represented, is finite only a finite subset S_0 of S is used. By the correctness of the calculus this says that S_0 is inconsistent. Part 1 of the lemma states that there a counterexamples to this, i.e., there is at least one infinite inconsistent set all of whose finite subsets are consistent. Thus the assumption of a sound and complete calculus for $\mathbf{DL}_{reg}$ has been refuted.

∎

An Infinitary Axiomatization

Definition 120 (The Calculus DL_{INF})

Axioms

Axioms for first-order Logic

Axioms for PDL

$\langle x := t\rangle F \quad \leftrightarrow \quad F[x/t]$ *for all first-order* F

Rules

$$\frac{F,\ F \to G}{G} \qquad \textit{(modus ponens)}$$

$$\frac{F}{[\pi]F} \qquad \frac{F}{\forall x F} \qquad \textit{(generalisations)}$$

$$\frac{G \to [\pi^n]F \textit{ for all } n}{G \to [\pi^*]F} \qquad \textit{for any first-order formula } F \textit{ (infinitary convergence)}$$

Theorem 109 *For any formula* F

$$F \textit{ is a tautology } \textit{ iff } \vdash_{INF} F$$

Here $\vdash_{\mathrm{INF}} F$ is used to denote the fact that F is derivable from the axioms and rules of the calculus DL_{INF}. For a proof see [Harel, 1984]. The reason why Theorem 109 does not contradict Lemma 108 lies in the fact that the last rule of DL_{INF} requires infinitely many premisses. A proof in DL_{INF} is thus no longer a finite object. This limits the usefulness of Theorem 109. It is an interesting theoretical insight that pinpoints the reason for non-axiomatisabilty in a classical proof system but it cannot be taken as a basis for automatic theorem proving.

Arithmetic Completeness

Also the next axiom system, that we call DL_{ARITH}, stretches the usual notion of an axiomatization beyond its limits. Derivations in DL_{ARITH} are simple string manipulations that can easily be automated except for the first set of axioms. The question which first-order formulas are true in $\mathbb{N}$ is undecidable.

Definition 121 (The Calculus DL_{ARITH})

Axioms

All first-order formulas valid in $\mathbb{N}$

Axioms for PDL

$\langle x := t\rangle F \quad \leftrightarrow \quad F[x/t]$ *for all first-order* F

Rules

$$\frac{F,\ F \to G}{G} \qquad \textit{(modus ponens)}$$

$$\frac{F}{[\pi]F} \qquad \frac{F}{\forall x F} \qquad \textit{(generalisations)}$$

$$\frac{\forall n(F(n+1) \to \langle\pi\rangle F(n))}{\forall n(F(n) \to \langle\pi^*\rangle F(0))} \qquad \begin{array}{r}\textit{for all first-order } F\\ n \textit{ does not occur in } \pi\\ \textit{(convergence)}\end{array}$$

In [Harel *et al.*, 2000] completeness for this calculus is proved. To distinguish this concept of completeness from the usual one it is called *arithmetic completeness*. We will not reproduce this proof here. But, it is interesting to look at the central part of this argument. The crucial lemma is:

Lemma 110 (Coding Lemma)
For every DL formula F *there is a first-order formula* F_L *such that*

$$(\mathcal{K}_{\mathbb{N}}, s) \models F \textit{ iff } (\mathbb{N}, s) \models F_L$$

for any $s : \mathrm{Var} \to \mathbb{N}$ *with* $\mathcal{K}_{\mathbb{N}}$ *the Kripke structure with* $\mathbb{N}$ *as computational structure.*

One might wonder how Lemma 110 could ever be true. The following lemma and the ensuing example should at least lend some confidence towards its truth.

Lemma 111

1. *There are formulas* $first$ *and* snd *in the vocabulary of* $\mathbb{N}$ *such that:*

$$\mathbb{N} \models \forall a \forall b \exists k(\forall x(first(k,x) \leftrightarrow x = a) \wedge \forall x(snd(k,x) \leftrightarrow x = b))$$

We may paraphrase this claim as: For any two numbers a, b *there is a number* k *that can serve as a code for the ordered pair* $\langle a, b\rangle$ *and the formulas* $first(k,x)$, $snd(k,y)$ *can be used to decode the first, resp. second element from the code* k.

2. *There is a formula seq in the vocabulary of* $\mathbb{N}$ *such that for every* $n \in \mathbb{N}$ *and any sequence* $k_0, \ldots, k_{n-1}$ *there is* $k \in \mathbb{N}$ *satisfying for each* i, $0 \leq i < n$

$$\mathbb{N} \models \forall x(seq(k, i, x) \leftrightarrow x = k_i)$$

This says: For any number n *and any sequence* $k_0, \ldots, k_{n-1}$ *of* n *numbers there is number* k *acting as a code for this sequence and the formula* $seq(k, i, x)$ *can be used to decode the* i*-the member of the coded sequence from* k*. Note, that the formula seq is uniform for all* n*.*

Proofs The proofs consist mostly of tedious computations. For (2) we can use the following

$$\begin{array}{lcl} k & = & \frac{1}{2}((a+b)(a+b+1))+a \\ first(u,x) & \equiv & \exists z(u = \frac{1}{2}((x+z)(x+z+1))+x \\ snd(u,x) & \equiv & \exists z(u = \frac{1}{2}((z+x)(z+x+1))+z \end{array}$$

Now, it takes only basic arithmetic and some concentration to complete the proof. For (2) we refer to the literature.

∎

The following example gives an idea how these encodings can be used to prove Lemma 110. We proceed in two phases. In the first we act as if there was a decoding function $seq(k, i)$ whose value is the i-th member of the sequence coded by k. But, Lemma 111 does not provide such a function, only the predicate defining its graph. In the second phase the function $seq(k, i)$ is replaced by the predicate $seq(k, i, x)$.

Example 34 (Example to Coding Lemma)
We will compute the first-order coding F_L *of the dynamic logic formula* $F(x) \equiv \langle (x > 0)?; x := x - 1)^* \rangle\ x = 0$.

$$
\begin{array}{rcl}
F_L(x) & \equiv & \exists n \exists k (x = seq(k,0) \ \wedge 0 = seq(k,n) \ \wedge \\
& & \forall i (0 \leq i < n \rightarrow seq(k,i) > 0 \ \wedge \\
& & seq(k,i+1) = seq(k,i)) - 1)) \\
& \equiv & \exists n \exists k (x = seq(k,0) \ \wedge \\
& & 0 = seq(k,n) \ \wedge \\
& & \forall i (0 \leq i < n \rightarrow \\
& & \rightarrow seq(k,i) > 0 \ \wedge \\
& & seq(k,i+1) = seq(k,i)) - 1)) \\
& \equiv & \exists n \exists k (\forall z (seq(k,0,z) \rightarrow x = z) \ \wedge \\
& & \forall z (seq(k,n,z) \rightarrow 0 = z) \ \wedge \\
& & \forall i \forall u \forall w (0 \leq i < n \wedge seq(k,i,u) \wedge seq(k,i+1,w) \\
& & \rightarrow u > 0 \ \wedge w = u - 1))
\end{array}
$$

Example 35 (Example Derivation)
We reuse the formula $F_L(x)$ from Example 34. We write for the purposes of the derivation $F_L = \exists n F_0$

$$
\begin{array}{lll}
1 & \vdash_{\mathbb{N}} \exists n F_0(n) & \textit{since true in } \mathbb{N} \\
2 & \vdash_{\mathbb{N}} F_0(0) \rightarrow x = 0 & \textit{since true in } \mathbb{N} \\
3 & \vdash_{\mathbb{N}} \forall n (F_0(n+1) \rightarrow \langle \alpha \rangle F_0(n)) & \textit{since true in } \mathbb{N} \\
4 & \vdash_{\mathbb{N}} \forall n (F_0(n) \rightarrow \langle \alpha^* \rangle F_0(0)) & \textit{by convergence rule from } 3 \\
5 & \vdash_{\mathbb{N}} \exists n (F_0(n)) \rightarrow \langle \alpha^* \rangle F_0(0) & \textit{first-order tautology} \\
6 & \vdash_{\mathbb{N}} \langle \alpha^* \rangle F_0(0) & 1, 5 \textit{ and modus ponens} \\
7 & \vdash_{\mathbb{N}} \langle \alpha^* \rangle x = 0 & 2, 6 \textit{ and modus ponens}
\end{array}
$$

with $\alpha \equiv x > 0?; x := x - 1$.

As can be seen in the last example the crucial part of the proof consists in convincing oneself that $\mathbb{N} \models \exists n F_0(n)$, $\mathbb{N} \models F_0(0) \rightarrow x = 0$ and $\mathbb{N} \models \forall n (F_0(n+1) \rightarrow \langle \alpha \rangle F_0(n))$ are true. This is a number theoretic problem instead of a proof theoretical one. The completeness proof is just an add-on, the crucial fact is contained in Lemma 110.

4.7 Exercises

Exercise 4.7.1
There is an intuitive feeling that the PDL Kripke structures from Definition 108 on page 179 are a generalisation of the ordinary Kripke structures as definied in Definition 37 on page 75 . This exercise will turn this intuition into a formal statement.

Let $\mathcal{K} = (S, \models, \rho)$ be a PDL Kripke structure and $a \in \mathrm{AP}$ an atomic program. We define the ordinary Kripke structure $\mathcal{K}^a = (G, R, v)$ by

1. $G = S$,
2. $R(g_1, g_2) \Leftrightarrow (g_1, g_2) \in \rho(a)$,
3. $v(g, p) = 1 \Leftrightarrow g \models p$.

Next let $F \in PFml$ be a PDL formula such that all modal operators occuring in F are of the form $[a]$ or $\langle a \rangle$. If we replace $[a]$ by $\Box$ and $\langle a \rangle$ by $\Diamond$ we arrive at an ordinary modal formula F^a. Prove for all $g \in G$

$$(\mathcal{K}, g) \models F \quad \Leftrightarrow \quad (\mathcal{K}^a, g) \models F^a$$

Exercise 4.7.2
Show that **repeat** α **until** $A \equiv \alpha; (\neg A?; \alpha)^*; A?$

Exercise 4.7.3
Show that the test operator ? (see Definition 96 on page 163) in regular Dynamic Logic can be expressed using **if** $-$ **then** $-$ **else**.

Exercise 4.7.4 *Show that the following DL formulas are tautologies for any program p.*

1. $\neg\langle p \rangle F \leftrightarrow [p]\neg F)$
2. $\neg[p]F \leftrightarrow \langle p \rangle \neg F)$
3. $[p](F \to G) \to (([p]F) \to [p]G)$

These are very elementary tautologies, valid in almost any modal logic. The proofs require little more then remembering the definitions.

Exercise 4.7.5
Let π be a PDL program. A pair of states (s_1, s_2) of a Kripke structure $\mathcal{K}$ is called $\langle\pi\rangle F$-accesible if either $s_1 \not\models \neg\langle\pi\rangle F$ or $s_2 \models \neg F$. This terminology is only used for this exercise.
Show that a pair of states (s_1, s_2) is $\langle\pi\rangle F$-accessible iff

$$\text{if } s_1 \models [\pi]\neg F \text{ then } s_2 \models \neg F$$

Exercise 4.7.6 (Deduction Theorem for PDL)
For PDL formulas G, F

$$G \vdash F \Leftrightarrow \quad \vdash G \to F$$

Exercise 4.7.7
The claim to be formulated in this exercise is the PDL analogon of the result presented in Exercise 3.9.13 for modal logic.
Let $\mathcal{K} = (S, \models, \rho)$ be a PDL Kripke structure, $A = \{a_1, \ldots, a_n\} \subseteq AP$ a finite set of atomic programs, and $s_0 \in S$. For the purposes of this exercise we call a program π an A-program and a formula $F \in PFml$ an A-formula if for any atomic program occuring in somewhere in π respectively in F we have $a \in A$.
Define inductively

$$\begin{array}{lcl} S^0_{s_0} & = & \{s_0\} \\ S^{n+1}_{s_0} & = & S^n_{s_0} \cup \{s_2 \mid (s_1, s_2) \in \rho(a_i) \text{ for some } s_1 \in S^n_{s_0} \text{ and } 1 \leq i \leq n\} \\ S^A_{s_0} & = & \bigcup_{0 \leq n} S^n_{s_0} \end{array}$$

Next we set $\mathcal{K}^A_{s_0} = (S^A_{s_0}, \models^A_{s_0}, \rho^A_{s_0})$ with

$$\begin{array}{lcl} \models^A_{s_0} & = & \models \cap (S^A_{s_0} \times \mathrm{PVar}) \\ \rho^A_{s_0}(a_i) & = & \rho(a_i) \cap (S^A_{s_0} \times S^A_{s_0}) \end{array}$$

Prove:

1. *For any $s \in S^A_{s_0}$, any A-program π, and $(s, s') \in \rho(\pi)$ we have $s' \in S^A_{s_0}$.*
2. *For any A-formula F and $s \in S^A_{s_0}$*

$$(\mathcal{K}, s) \models F \quad \Leftrightarrow \quad (\mathcal{K}^A_{s_0}, s) \models F$$

Exercise 4.7.8
Here the reader is asked to prove a version of a modified deduction theorem for the global consequence relation for PDL. Let $A = \{a_1, \ldots, a_n\} \subseteq \mathrm{AP}$ *be a finite set of atomic programs and* F_1, F_2 *A-formulas (see Exercise 4.7.7 for this terminology). Then*

$$F_1 \vdash_G F_2 \quad \Leftrightarrow \quad \vdash [(a_1 \cup \ldots \cup a_n)^*]F_1 \to F_2$$

Exercise 4.7.9 *Show that the satisifability problem for the modal logic* **K** *(see Definitions 40 and 41) is decidable.*
Hint: Try to reduce the problem to Theorem 103.

Exercise 4.7.10 *Show that the satisifability problem for the modal logic* **S4** *with reflexive, transitive accessability relation (see Definitions 40 and 41) is decidable.*
Hint: Try to reduce the problem to Theorem 103.

Exercise 4.7.11 *Because of the deduction theorems, see Theorem 54 for modal logic and Exercise 4.7.6 for PDL, also the logical consequence* $F_1 \vdash F_2$ *is decidable for both logics.*
What about the global consequence relation $F_1 \vdash_G F_2$*?*
Show that also $F_1 \vdash_G F_2$ *is decidable for modal logic.*
Hint: use Exercise 3.9.14.

Exercise 4.7.12 *Show that* $F_1 \vdash_G F_2$ *is decidable also for PDL.*
Hint: use Exercise 4.7.8.

Chapter 5

Temporal Logics

This chapter on Temporal Logics starts in Section 5.1 with automata on infinite words, also called omega words. These automata are called Büchi automata to honour the Swiss mathematician Julius Richard Büchi who invented them in 1962. Büchi automata extend the well-known concept of finite-state automata. The close connection between finite-state automata and regular sets of (finite) words is paralleled by the correspondence between Büchi automata and omega-regular sets of omega words. We will sometimes use the greek letter ω instead of *omega*. The presentation in these lecture notes assumes that the reader is already familiar with Büchi automata and omega regular sets and thus only presents a quick review of the central definitions and facts. The main new results is Theorem 114 on the equivalence of Büchi acceptance and definability in second-order logic.

There are many reasons why one might want to study omega words. Our perspective in this lecture notes is on the formal analysis of discrete systems. We do not want to enter a general overview on discrete systems, but we want to offer some remarks on how the material covered in the following sections fits into a more global context. In particular these remarks should help to appreciate the next theorem. A discrete system starts in some initial state s_0 and proceeds through a series of steps $s_0, s_1, \ldots, s_i, \ldots$. Here the s_i are taken from a finite set of possible system states. This scenario exludes continuous systems that are instead desribed by functions on the real numbers. The analysis we have in mind usually takes place at the level of abstraction where not all details of the system under investigation are taken into account. This is one reason why we look at non-deterministic systems. Thus

$s_0, s_1, \ldots, s_i, \ldots$ is just one path, or computation sequence of the systems, there might be others, e.g., $s_0, t_1, \ldots, t_i, \ldots$. There is no limit to what kind of analysis one might want to perform. Just to have some example in the back of your mind while reading on we mention the two questions: Is it possible that the system starting in s_0 will reach a state s with a certain property P_1? or will in all reachable states of the system the property P_2 be true? This necessitates that we have some means to express properties of states. To keep things simple we will only consider properties that can be expressed in propositional logic. This is by far the most commonly investigated case. Consequently, we need the information which propositional atoms are true in which states. The concept of an omega structure, see Definition 131, formalizes this concept of a path in a system or computation sequence. The approach described so far leads to Linear Temporal Logic, LTL, that will be reviewed in Section 5.2. The semantics of LTL so far was only formulated with respect to one given path or omega structure. We stipulate that an LTL formula is true for a discrete system, i.e., for the set of paths given by the system, if the formula is true for all computation sequences. With this definition it is not possible to formulate statements about the *existence* of a computation with certain properties. This leads to the introduction of Computation Tree Logic, CTL, in Section 5.4 that allows quantification over computation paths.

There are many ways to describe the allowed steps or transitions from one state s to the next state. The simplest way is to give for every s a set of possible successor states, this leads to the concept of a Kripke structure, or of a simple transition system as given later on in Definition 138. More elaborate descriptions of state transitions could include a guard formula that needs to be statisfied for the transition to be activated. The description might also include an action part that states what properties should hold in the post state(s). Another possibility is to put labels on transitions. Since the vocabulary for labels is completely arbitrary this is a very powerful mechanism. Labels are used in finite automata or Büchi automata. In the case of finite automaton the states do not carry additional information. There are no propositional variables that can be used to express properties of states. The typical questions one would ask about automata of this kind would be: Is there a sequence of labels such that a particular state, e.g., a final state, is reached?

Each of these possibilities has its own merits and draw-backs and its is fre-

quently possible to mimick one approach by another. One could e.g., compensate the lack of guards on transitions in Kripke structures by adding additional states.

The rich theory of LTL and CTL already make these logics an indispensible topic in any curriculum of theoretical computer science. The full importance of these logics however can only be appreciated after the presentation of the corresponding model checking procedures. Given a formal description of a system model $\mathcal{M}$ and an LTL or CTL formula A highly efficient model checking algorithms can be used to determine fully automatically wether property A holds true in the model $\mathcal{M}$. Since LTL model checking has already been covered in the Formal Systems 1 course Section 5.3 describes an alternative approach: translating an LTL model checking task into a propositional satisfiability problem. Section 5.5 is devoted to model checking of CTL properties.

5.1 Büchi Automata

We assume that the reader is familiar with the basic concepts of omega-words and Büchi automata. For the ease of the reader we repeat the crucial definitions here. Further information may be found in [Schmitt, 2008].

Definition 122 (Omega words)
Let V be a finite alphabet.
V^ω denotes the set of infinite words made up of letters from V.
In contrast V^* denotes the set of all finite words over V.
For $n \geq 0$ the notation $w(n)$ refers to the n-th letter in w and $w \downarrow (n)$ stands for the finite initial segment $w(0) \dots w(n)$ of w.
Omega words are sometimes also called infinite words or referenced as ω words using the last letter of the Greek alphabet.

Definition 123 (Operations on omega words)
Let $K \subseteq V^*$ be a set of finite words and $J \subseteq V^\omega$ a set of infinite words.

1. *K^ω denotes the set of omega words*

$$w_1 \dots w_i \dots \text{ such that } w_i \in K \text{ for all } i$$

2. *The concatenation of K with J is defined by*

$$KJ = \{w_1 w_2 \quad | \quad w_1 \in K, w_2 \in J\}$$

   Note, that concatenation of two sets of infinite words does not make sense.

3. $$\vec{K} = \{w \in V^\omega \mid w \downarrow (n) \in K \text{ for infinitely many } n\}$$

   Since we may think of $\vec{K}$ as a kind of limit of K (in the mathematical sense of the word) some authors denote it by $lim(K)$.

Definition 124 (Büchi Automaton)
A Büchi automaton $\mathcal{A} = (S, V, s_0, \delta, F)$ is a non-deterministing finite automaton with

S	*a finite set of states*
V	*an alphabet*
$s_0 \in S$	*the initial state*
$\delta : S \times V \to \mathcal{P}(S)$	*the transition function*
$F \subseteq S$	*the set of accepting (or final) states*

A run *(also called a computation sequence) for $\mathcal{A}$ is an infinite sequence $s_0, \ldots, s_n, \ldots$ of states, starting with s_0 such that for all $0 \leq n$ there is an $a \in V$ with $s_{n+1} \in \delta(s_n, a)$.*
A run $(s_n)_{n \geq 0}$ is called an accepting run if $s_n \in F$ is true for infinitely many n.

Given an omega-word $w \in V^\omega$ we call a run $s_0, \ldots, s_n, \ldots$ a run for w, if for all $0 \leq n$

$$s_{n+1} \in \delta(s_n, w(n))$$

is true. The language $L^\omega(\mathcal{A})$ accepted by $\mathcal{A}$ is defined by

$$L^\omega(\mathcal{A}) = \{w \in V^\omega \mid \text{there is an accepting run for } w\}$$

The sets L of omega-words accepted by a Büchi automaton $\mathcal{A}$, i.e., $L^\omega(\mathcal{A}) = L$ are also called omega-regular *sets.*

Note, that a Büchi automaton $\mathcal{A}$ is nothing else but a well known nondeterministic finite automaton. It is the acceptance condition that makes all the difference.

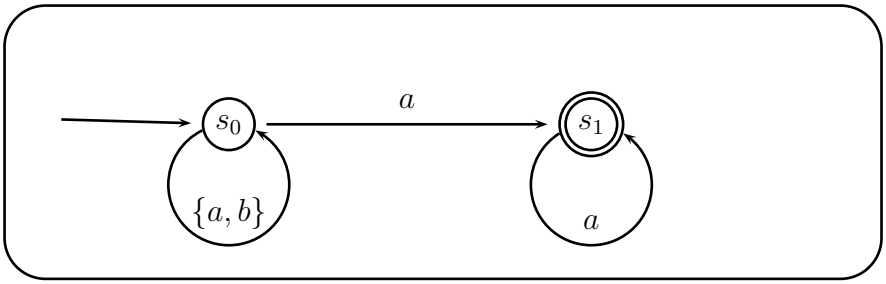


Figure 5.1: The Büchi automaton $\mathcal{N}_{afin}$

We usually present the small examples of automata in this lecture notes graphically as in Figure 5.1. It can be easily seen that $L^\omega(\mathcal{N}_{afin})$ is the set

of all omega words that eventually are equal to a. Using the operators from Definition 123 we may write $L^\omega(\mathcal{N}_{afin}) = \{a, b\}^* a^\omega$

Second-Order Definability

So far we have modelled omega words as functions from $\mathbb{N}$ into the alphabet V. We will now present another, less obvious, way to represent omega words. We will consider them as structures for a logic with the following vocabulary Σ_V.

Definition 125 (Σ_V)
Let V be a finite alphabet. The vocabulary Σ_V consists of the following functions and relation symbols:

a constant symbol	0	*the first element*
a unary function symbol	s	*the successor function*
a unary relation symbol	$a(x)$	*for any letter a in V*
a binary relation symbol	$<$	*the order relation*

If $\mathcal{W}$ is a structure in this vocabulary we think of the elements in the universe W of $\mathcal{W}$ as the positions of a word. $0^{\mathcal{W}}$ will be the first position and $s^{\mathcal{W}}$ will yield for any position the next position. $\mathcal{W} \models a[p]$ for a position $p \in W$ is supposed to mean that at p the letter $a \in V$ occurs. Of course not every Σ_V structure makes sense. We need to make sure that

1. 0 is interpreted as the smallest element.
2. For every element exactly one of the atomic formulas $a(x)$ for $a \in V$ is true.
3. For all elements p $s(p)$ is the successor of p.
4. The successor s is injective.
5. $<$ is a transitive and irreflexive relation.
6. All elements are of the form $s^n(0)$ for some $n \in \mathbb{N}$.

Except item 6 these properties could be expressed as formulas in first-order logic. But, property 6 requires that we use second order logic, more precisely we will use the monadic second order logic introduced in Section 3.4. Let us denote the conjunction of the following monadic second-order formulas by FOW:

1. $\forall x(0 = x \vee 0 < x)$
2. $\forall x \bigvee_{a \in V}(a(x) \wedge \bigwedge_{a,b \in V, a \neq b} \neg(a(x) \wedge b(x))$
3. $\forall x(x < s(x)) \wedge \neg\exists z(x < z \wedge z < s(x))$
4. $\forall x \forall y(s(x) = s(y) \rightarrow x = y)$
5. $\forall x \forall y \forall z(x < y \wedge y < z \rightarrow x < z) \wedge \forall x \neg(x < x)$
6. $\forall X(X(0) \wedge \forall x(X(x) \rightarrow X(s(x))) \rightarrow \forall y(X(y)))$

Lemma 112
There is a 1-1 correspondence between omega words over the alphabet V and Σ_V structures satisfying FOW.
For $w \in V^\omega$ we will denote the correspondig structure by $\mathcal{W}_w$. Without loss of generality we will assume that the universe of $\mathcal{W}_w$ is the set $\mathbb{N}$ of natural numbers.

Proof Simple.

∎

The following lemma will be useful to shorten the proof of Theorem 116 below.

Lemma 113 (Definability of the order relation)
The following formula

$$less(x, y) = \forall X(X(x) \wedge \forall z(X(z) \rightarrow X(s(z))) \rightarrow X(y)) \wedge x \neq y$$

defines the order relation in all structures satisfying FOW.

Proof Let $\mathcal{W}$ be a model of FOW. Without loss of generality we may assume that the universe W of $\mathcal{W}$ is $\mathbb{N}$ and $s^{\mathcal{W}}(n) = n + 1$. For $n \in \mathbb{N}$ we define the set $N_n = \{n' \mid n < n'\}$. If $less^{\mathcal{W}}(n, m)$ then $n \neq m$ and $m \in N_n$, i.e., $n < m$. The other way round if $n < m$ is true and S is a subset with $n \in S$ and for all n' $n' \in S \rightarrow (n' + 1) \in S$ then we must also have $m \in S$. This shows $less^{\mathcal{W}}(n, m)$.

▪

The representation of infinite words from Lemma 112 opens the possibility to define sets $L \subseteq V^\omega$ by formulas F in monadic second order logic. Consider the formula $F = \exists n \forall m(n < m \rightarrow a(m))$. The set of words defined by F, let us denote it by $L^\omega(F)$, equals the set of words $L^\omega(\mathcal{N}_{afin})$ defined by the Büchi automaton in Figure 5.1. Spelled out in full, this means that for any $w \in V^\omega$ the structure $\mathcal{W}_w$ satisfies the formula F (i.e., $\mathcal{W}_w \models F$) iff $w \in L^\omega(\mathcal{N}_{afin})$. We will use the short version from now on. Whenever we talk about monadic second order formulas we mean second order formulas in the signature Σ_V.

Theorem 114 (Charactization of Büchi acceptance)
Let $L \subseteq V^\omega$ be a set of infinite words.
There is a Büchi automaton accepting L iff there is a second order formula in signature Σ_V defining L.

Since the proofs for both directions of Theorem 114 are quite involved we split this theorem into Theorem 115 and Theorem 116.

Theorem 115 (Second Order Definability Büchi acceptance)
Let $L \subseteq V^\omega$ be a set of infinite words.
If there is a Büchi automaton $\mathcal{N}$ accepting L, i.e., $L^\omega(\mathcal{N}) = L$, then there is a second order formula Φ defining L. i.e., $L^\omega(\Phi) = L$.

Proof This is the easier implication in the equivalence claim of Theorem 114. We start from a given Büchi automaton $\mathcal{N} = (S, V, s_0, \delta, F)$. Let $S = \{q_1, \ldots, q_n\}$ with $F = \{q_1, \ldots, q_f\}$. Let us carefully examine the definition of $w \in L^\omega(\mathcal{N})$:

- there is a run $s = (s_i)_{0 \leq i}$ of $\mathcal{N}$ for w with infinitely many final states.

The essential trick is to rephrase this as follows:

- there are finitely many subsets $X_k \subseteq \mathbb{N}$, $1 \leq k \leq n$ such that
  1. $\bigcup_{0 \leq k} X_k = \mathbb{N}$
  2. the X_k are mutually disjoint
  3. the sequence of states s_i definied by

$$s_i = q_k \quad \Leftrightarrow \quad i \in X_k$$

  is a run of $\mathcal{N}$ for w with infinitely many final states.

Now we are in bussiness.

$$\Phi = \exists X_1 \ldots \exists X_n (U \wedge D \wedge R \wedge F)$$

with

$$\begin{array}{lcl} U & = & \forall x (\bigvee_{1 \leq k \leq n} X_k(x)) \\ D & = & \bigwedge_{1 \leq k < r \leq n} \neg \exists x (X_k(x) \wedge X_r(x)) \\ R & = & \forall x (\bigwedge_{1 \leq k,r \leq n} (X_k(x) \wedge X_r(s(x)) \rightarrow \bigvee_{a \in V_{k,r}} a(x))) \\ & & \text{with } V_{k,r} = \{a \in V \mid \delta(q_k, a) = q_r\} \\ F & = & \bigvee_{1 \leq k \leq f} \forall x \exists y (x < y \wedge X_k(y)) \end{array}$$

It should be clear that this $\mathcal{W}_w \models \Phi$ exactly if $w \in L^\omega(\mathcal{N})$.

∎

Theorem 116 (Omega Regularity of Definable Sets)
For any second order formula Φ in the signature Σ_V the set $L^\omega(\Phi) = L$ is omega regular.

Proof Be prepared, this proof will take a while.

Naturally, the proof will proceed by induction on the complexity of Φ. This necessitates that we also consider formulas $\Phi(x_1, \ldots, x_p, X_1, \ldots, X_q)$ with free first-order and second-order variables. What is the claim of the theorem in this case? We will for any numbers p of first-order and q of second-order free variables define new vocabularies $V_{p,q}$ such that $V_{0,0} = V$ and $\Phi(x_1, \ldots, x_p, X_1, \ldots, X_q)$ defines an omega-regular subset of $V^\omega_{p,q}$.

As a first step of filling in the details of this new claim we introduce a more convenient notation of relating free variables to their assignments.

Instead of $val_{\mathcal{M},\beta,\gamma}(\Phi(x_1,\ldots,x_p,X_1,\ldots,X_q)) = 1$, we will write $\mathcal{M} \models \Phi(x_1,\ldots,x_p,X_1,\ldots,X_q)[s_1,\ldots,s_p,S_1,\ldots,S_q]$ with $s_i = \beta(x_i)$ and $S_j = \gamma(X_j)$. Postfixing the assignments in square brackets to the formula is more convenient than manipulating the βs and γs and in our present restricted context there is no danger of confusing which element or which set gets assigned to which variable.

Definition 126 ($V_{p,q}$) *The letters in $V_{p,q}$ are sequences of length $p+q+1$ of letters from V in the first position and 0 and 1 in the rest. More precisely*

$$V_{p,q} = V \times \{0,1\}^p \times \{0,1\}^q$$

For $p = q = 0$ we get $V_{p,q} = V$, as promissed. As an example consider $p = 3$, $q = 2$ and $a \in V$. Then

$$\begin{array}{rcl} b_1 & = & \langle a,0,0,0,0,0\rangle \\ b_2 & = & \langle a,0,0,1,0,0\rangle \\ b_3 & = & \langle a,0,1,0,0,1\rangle \end{array}$$

are all letters in $V_{3,2}$.

The subset $K_{p,q} \subseteq V_{p,q}^\omega$ will play a crucial role in what follows.

Definition 127 ($K_{p,q}$) *The subset $K_{p,q} \subseteq V_{p,q}^\omega$ is defined by*

$$K_{p,q} = \{w \in V_{p,q}^\omega \mid \text{for all } 1 \le i \le p \text{ there is exactly one } k \text{ such that the } i\text{-th position in the letter } w(k) \text{ equals } 1\}$$

If we use the notation $a[i]$ for $1 \le i \le p+q+1$ to denote the i-th position of a letter $a \in V_{p,q}$ we can rewrite the definition of $K_{p,q}$ as

$$K_{p,q} = \{w \in V_{p,q}^\omega \mid \text{for all } 1 \le i \le p \text{ there is exactly one } k \text{ with } w(k)[i] = 1\}$$

Observe that $K_{0,0} = V$. It is left as an intext exercise to the reader to convince herself/himself that $K_{p,q}$ is omega-regular.

A word $w \in K_{p,q}$ codes a word $w^0 \in V^\omega$, p elements $s_{w,i} \in \mathbb{N}$ and q subsets $S_{w,j} \subseteq \mathbb{N}$:

$$\begin{array}{lcll} w^0(n) & = & w(n)[1] & n \in \mathbb{N} \\ s_{w,i} & = & \text{the unique } n \text{ with } w(n)[1+i] = 1 & 1 \le i \le p \\ S_{w,j} & = & \{n \in \mathbb{N} \mid w(n)[1+p+j] = 1 & 1 \le j \le q \end{array}$$

The claim we will prove by structural induction on Φ can now be stated as

$$\begin{array}{l}\text{For any formula} \\ \qquad \Phi(x_1, \ldots, x_p, X_1, \ldots, X_q) \\ \text{the set} \\ L^\omega(\Phi(\bar{x}, \bar{X})) = \{w \in K_{p,q} \mid \mathcal{W}_{w^0} \models \Phi[s_{w,1}, \ldots, s_{w,p}, S_{w,1}, \ldots, S_{w,q}]\} \\ \text{is omega regular}\end{array} \tag{5.1}$$

$\mathcal{W}_{w^0}$ is the structure from Lemma 112. Also note that the special case of $p = q = 0$ is the claim of the theorem we want to prove.

The inductive proof of claim 5.1 starts with considering the atomic and negated atomic formulas Φ. Taking into account the definability result from Lemma 113 we need to consider

$$x_i = x_k, x_i = s(x_k), a(x_i), X_j(x_i)$$

We consider the cases $x_i = x_k$.
Let $A_{i,k} \subseteq V_{p,q}$ be the set of all letters a with $a[1+i] = a[1+k] = 0$ and $B_{i,k} \subseteq V_{p,q}$ be the set of all letters b with $a[1+i] = a[1+k] = 1$. Then with the notation used in claim 5.1

$$\begin{array}{rcl} L^\omega(x_i = x_k) & = & \{w \in K_{p,q} \mid \mathcal{W}_{w^0} \models x_i = x_k[s_i, s_k]\} \\ & = & K_{p,q} \cap A_{i,k}^* B_{i,k} A_{i,k}^\omega \end{array}$$

Using the results from [Schmitt, 2008] we see that $L^\omega(x_i = x_k)$ is omega regular.

The case $x_i = s(x_k)$ is only a minor variation of this proof.

Let $D_{a,i}$ be the set of letters $d \in V_{p,q}$ with $d[1] = a$ and $d[1+i] = 1$. Then

$$\begin{array}{rcl} L^\omega(a(x_i)) & = & \{w \in K_{p,q} \mid \mathcal{W}_{w^0} \models a(x_i)[s_i]\} \\ & = & \text{for the unique } n \text{ with } w(n)[1+i] = 1 \text{ we must have } w(n)[1] = a \\ & = & K_{p,q} \cap V_{p,q}^* D_{a,k} V_{p,q}^\omega \end{array}$$

Finally let $E_{i,j} = \{e \in V_{p,q} \mid e[1+i] = 1 \wedge e[1+p+j] = 1\}$ then

$$L^\omega(X_j(x_i)) \quad = \quad K_{p,q} \cap V_{p,q}^* E_{i,j} V_{p,q}^\omega$$

In all cases we see that omega regular sets arise.

The inductive steps $\Phi = \neg\Phi_1$ $\Phi = \Phi_1 \wedge \Phi_2$ and $\Phi = \Phi_1 \vee \Phi_2$ follow immediately form the fact that omega regular sets are closed under complement, intersection and union, see e.g., [Schmitt, 2008]. It remains to consider first order and second order quantification. It suffices to consider existential quantifiers.

$\Phi(\bar{x}, \bar{X}) = \exists x_i \Phi_0(\bar{x'}, \bar{X})$. To avoid tedious manipulation of indices we assume that $i = p$. Thus $\Phi(x_1, \ldots, x_{p-1}, \bar{X}) = \exists x_p \Phi_0(x_1, \ldots, x_{p-1}, x_p, \bar{X})$. By induction hypothesis we know that $L_1 = \{w \in K_{p,q} \mid \mathcal{W}_{w^0} \models \Phi_0(\bar{x}, x_p, \bar{X})[\bar{s}, s_p, \bar{S}]\}$ is an omega regular subset of $V^\omega_{p,q}$. We want to show that $L = \{w \in K_{p-1,q} \mid \mathcal{W}_{w^0} \models \exists x_p \Phi_0(\bar{x}, \bar{X})[\bar{s}, s_p, \bar{S}]\}$ is an omega regular subset of $V^\omega_{p-1,q}$. Relating subsets over different vocabularies is something we have not encountered before. We consider the mapping $\mu : V_{p,q} \to V_{p-1,q}$ that maps letters from $V_{p,q}$ to letters in $V_{p-1,q}$ be simply dropping the $1 + p$-th position, i.e.,

$$\mu((a, c_1, \ldots, c_{p-1}, c_p, d_1, \ldots, d_q)) = (a, c_1, \ldots, c_{p-1}, d_1, \ldots, d_q)$$

For $w \in V^\omega_{p,q}$ we denote by $\mu(w) \in V^\omega_{p-1,q}$ the word that arises from w by replacing each letter c by $\mu(c)$.

We will first show that $L = \{\mu(w) \mid w \in L_1\} = \mu(L_1)$. To increase readability we supress all variables except x_p.

$$\begin{array}{lll}
w \in L_1 & \text{iff} & \mathcal{W}_{w^0} \models \Phi_1(x_p)[s_p] \\
 & \Rightarrow & \mathcal{W}_{w^0} \models \exists x_p \Phi_1 \\
 & \text{iff} & \mathcal{W}_{\mu(w)^0} \models \exists x_p \Phi_1 \qquad \mu(w)^0 = w^0 \text{ and also the assigments of the not shown free variables are the same for } w \text{ and } \mu(w) \text{ since } \mu \text{ only touches position } p \\
 & \text{iff} & \mu(w) \in L
\end{array}$$

$$\begin{array}{lll}
w' \in L & \text{iff} & \mathcal{W}_{(w')^0} \models \exists x_p \Phi_1 \\
 & \text{iff} & \mathcal{W}_{w^0} \models \exists x_p \Phi_1 \\
 & \text{iff} & \mathcal{W}_{(w')^0} \models \Phi_1(x_p)[n_p] \qquad \text{for some } n_p \in \mathbb{N} \\
 & \text{iff} & \mathcal{W}_{w^0} \models \Phi_1(x_p)[s_p] \qquad \text{for an appropriate } w \in V^\omega_{p,q} \\
 & \text{iff} & w \in L_1
\end{array}$$

The appropriate w is obtained by replacing the letter $w'(n) = (a, c_1, \ldots, c_{p-1}, d_1, \ldots, d_q)$ by $\begin{cases} (a, c_1, \ldots, c_{p-1}, 1, d_1, \ldots, d_q) & \text{if } n = n_p \\ (a, c_1, \ldots, c_{p-1}, 0, d_1, \ldots, d_q) & \text{if } n \neq n_p \end{cases}$

Obviously $\mu(w) = w'$. Thus $L = \mu(L_1)$ is established. It remains to argue

that because L_1 is omega regular $\mu(L_1)$ is also. This is the content of the following lemma, whose simple proof is left as an exercise, see Exercise 5.6.7.

Lemma 117 (Homomorphic images of omega regular sets)
Let V_1, V_2 be two vocabularies, $\mu : V_1 \to V_2$ an arbitrary mapping and L an omega regular subset of V_1^ω.
Then $\mu(L)$ is an omega regular subset of V_2^ω.

The induction step from Φ to $\exists X_j \Phi$ can be proved along the same lines.

∎

5.2 Linear Temporal Logic

The linear temporal logic, LTL, belongs to the family of propositional modal logics.

Definition 128 (LTL Formulas)
Let PVar *be a set of propositional atoms. The set of LTL formulas, $LTLFor$ (if needed will will write more precisely $LTLFor_{\mathrm{PVar}}$) is defined by*

1. $\mathrm{PVar} \subseteq LTLFor$
2. $\mathbf{true}, \mathbf{false} \in LTLFor$
3. *If A, B in $LTLFor$, then also $\neg A$, $A \wedge B$, $A \vee B$, $A \to B$*
4. *if $A, B \in LTLFor$ then also*
   (a) $\mathbf{G}A$ *and* $\mathbf{F}A$,
   (b) $A\ \mathbf{U}\ B \in LTLFor$
   (c) $\mathbf{X}\ A$

It would be more consistent with a modal interpretation of temporal logic to write $\Box A$ and $\Diamond A$ instead of $\mathbf{G}A$ and $\mathbf{F}A$. But, this latter notion is more widespread. It may be easier to remember when you think of $\mathbf{F}$ refering to sometime in the future and $\mathbf{G}$ to mean globally true for all times to come.

Later we will need the maximal nesting depth $od(A)$ of temporal operators in a given LTL-formula A. Here is the precise definition.

Definition 129 (Operator Depth)

1. $od(A) = 0$ *for* $A \in \mathrm{PVar}$
2. $od(\mathbf{true}) = od(\mathbf{false}) = 0$
3. $od(\neg A) = od(A)$, $od(A \wedge B) = max\{od(A), od(B)\}$, *etc ...*
4. $od(\mathbf{G}A) = od(A) + 1$, $od(\mathbf{F}A) = od(A) + 1$,

5. $od(A\ \mathbf{U}\ B) = max\{od(A), od(B)\} + 1$

6. $od(\mathbf{X}\ A) = od(A) + 1$

Definition 130 (Temporal Structures)
A temporal structure is a triple $\mathcal{T} = (S, R, I)$ *with*

S	*a nonempty set, called the set of abstract time points.*
R	*a strict transitive relation thought of as representing the temporal* before *relation, we will use the more suggestive symbol* $<$ *instead of* R
I	*a valuation function* $I : (\mathrm{PVar} \times S) \to \{\mathbf{W}, \mathbf{F}\}$

Definition 131 (Omega Structures)
An omega structure (S, R, I) *is the special case of a temporal structure with* $(S, R) = (\mathbb{N}, <)$.

We will present omega structures in the equivalent form

$$\mathcal{T} = (\mathbb{N}, <, \xi)$$

with $\xi : \mathbb{N} \to 2^{\mathrm{PVar}}$ *and the intuition*

$$p \in \xi(n) \Leftrightarrow \text{ in } \mathcal{T} \text{ atom } p \text{ is true at time point } n$$

For $\xi : \mathbb{N} \to 2^{\mathrm{PVar}}$ *and* $n \in \mathbb{N}$ *we will use* ξ_n *to stand for the final segment of* ξ *starting at* n. *In symbol* $\xi_n(m) = \xi(n + m)$. *In particular* $\xi_0 = \xi$.

Definition 132 (LTL Semantics)
Let $\mathcal{T} = (\mathbb{N}, <, \xi)$ *be an omega structure of the propositional signature* PVar.

$\xi \models p$	*iff*	$p \in \xi(0)$ *(p an AL atom)*
$\xi \models op(A, B)$		*for a propositional combination* $op(A, B)$ *of* A *and* B *as usual*
$\xi \models \mathbf{G}A$	*iff*	*for all* $n \in \mathbb{N}$ *it is true* $\xi_n \models A$
$\xi \models \mathbf{F}A$	*iff*	*there exists an* $n \in \mathbb{N}$ *with* $\xi_n \models A$
$\xi \models A\ \mathbf{U}\ B$	*iff*	*there is* $n \in \mathbb{N}$ *with* $\xi_n \models B$ *and for alle* m *with* $0 \leq m < n$ *we have* $\xi_m \models A$
$\xi \models \mathbf{X}\ A$	*iff*	$\xi_1 \models A$

Lemma 118 (Simple LTL Tautologies)
The following formulas are true for all omega structures:

1. $\mathbf{F}A \leftrightarrow \mathbf{true}\ \mathbf{U}\ A$
2. $\mathbf{G}A \leftrightarrow \neg(\mathbf{true}\ \mathbf{U}\ \neg A)$

Lemma 118 shows that the logical constants **true** and **false** and the $\mathbf{U}$ operator suffice to express the other temporal operators. For convenience, clarity, or other reasons it is sometime useful to consider additional definable operators. We will study here $\mathbf{U}_w$ and $\mathbf{V}$. $\mathbf{U}_w$ is called the *weak until operator* while there is no agreed name for $\mathbf{V}$.

Definition 133 (Additional Operators)

$\xi \models A\ \mathbf{U}_w\ B$ *iff* $\xi_n \models (A \wedge \neg B)$ *holds for all* $n \in \mathbb{N}$ *or there exists an* $n \in \mathbb{N}$ *such that* $\xi_n \models B$ *and* $\xi_m \models A$ *for all* m *with* $0 \leq m < n$

$\xi \models A\ \mathbf{V}\ B$ *iff* $\xi \models B$ *and for all* $n \in \mathbb{N}$ *holds if* $\xi_n \models \neg B$ *then there is* m *such that* $0 \leq m < n$ *and* $\xi_m \models A$

Lemma 119 *For all omega structures the following equivalences are true:*

1. $A\ \mathbf{U}\ B \leftrightarrow A\ \mathbf{U}_w\ B \wedge \Diamond B$
2. $A\ \mathbf{U}_w\ B \leftrightarrow A\ \mathbf{U}\ B \vee \Box(A \wedge \neg B)$
3. $A\ \mathbf{V}\ B \leftrightarrow \neg(\neg A\ \mathbf{U}\ \neg B)$
4. $A\ \mathbf{U}\ B \leftrightarrow (B \vee (A \wedge \mathbf{X}\ (A\ \mathbf{U}\ B)))$
5. $A\ \mathbf{V}\ B \leftrightarrow (B \wedge A) \vee (B \wedge \mathbf{X}\ (A\ \mathbf{V}\ B))$

Proofs: See Formale Systeme lecture notes [Schmitt, 2008].

Comparing omega structures and Büchi automata we notice that the states in the automata model do not carry any information about propositional variables being true or false. This can be compensated by coding it into the label. This is exactly what happens in the next theorem. On one side, in fact on the left-hand side of the equation, sequences of labels of an Büchi automaton are considered. On the right-hand side there occur omega structures. Omega structures can be coded as sequences of labels, or as omega words, by choosing an appropriate label vocabulary. If PVar is the set of propositional variabels occuring in the omega structures then a letter in the label vocabulary V should be a subset of PVar. In total V is a subset of all subsets of PVar.

Theorem 120 *For every LTL formula C there is a Büchi automaton $\mathcal{A}_C$ such that*

$$L^\omega(\mathcal{A}_C) = \{\xi \in V^\omega \mid \xi \models C\}.$$

Proof See [Schmitt, 2008, Theorem 11.3].

One particularly interesting step in the proof of Theorem 120 is the inductive step where C is a conjunction $C_1 \wedge C_2$. This is handled by the next lemma.

Lemma 121 *Let $\mathcal{A}_1 = (S_1, V, s_1^0, \delta_1, F_1)$, $\mathcal{A}_2 = (S_2, V, s_2^0, \delta_2, F_2)$ be given Büchi automata, C_1, C_2 LTL formulas such that $L^\omega(\mathcal{A}_1) = \{\xi \in V^\omega \mid \xi \models C_1\}$ and*
$L^\omega(\mathcal{A}_2) = \{\xi \in V^\omega \mid \xi \models C_2\}$
Then there is a Büchi automaton $\mathcal{C}$ satisfying

$$L^\omega(\mathcal{C}) = \{\xi \in V^\omega \mid \xi \models (C_1 \wedge C_2)\}$$

We sometimes use the notation $\mathcal{C} = \mathcal{A}_1 \wedge \mathcal{A}_2$.

Proof See [Schmitt, 2008, Lemma 11.2].
In fact, the automaton $\mathcal{A}_1 \wedge \mathcal{A}_2$ can be explicitely defined by:

$$
\begin{array}{lcl}
S & = & S_1 \times S_2 \times \{1,2\} \\
s^0 & = & (s_1^0, s_2^0, 1) \\
F & = & F_1 \times S_2 \times \{1\} \\
 & & \text{for all } s_1 \in S_1, s_2 \in S_2, i \in \{1,2\} \\
(t_1,t_2,i) \in \delta^0((s_1,s_2,i),a) & \Leftrightarrow & t_1 \in \delta_1(s_1,a) \text{ and } t_2 \in \delta_2(s_2,a) \\
\text{if } s_1 \in F_1 & & \\
(t_1,t_2,2) \in \delta^1((s_1,s_2,1),a) & \Leftrightarrow & t_1 \in \delta_1(s_1,a) \text{ and } t_2 \in \delta_2(s_2,a) \\
\text{if } s_2 \in F_2 & & \\
(t_1,t_2,1) \in \delta^2((s_1,s_2,2),a) & \Leftrightarrow & t_1 \in \delta_1(s_1,a) \text{ and } t_2 \in \delta_2(s_2,a) \\
\delta((s_1,s_2,1),a) = \delta^0((s_1,s_2,1),a) \cup \delta^1((s_1,s_2,1),a) & & \\
\delta((s_1,s_2,2),a) = \delta^0((s_1,s_2,2),a) \cup \delta^2((s_1,s_2,1),a) & &
\end{array}
$$

■

We now turn our attention to an alternative of the semantics for LTL formulas as given in Definition 132. This alternative, which is in fact only a small shift in the point of view, is motivated by applications of LTL, where one is interested in proving properties of system models. Typically a system may be modelled by a Büchi automaton. The statement, a Büchi automaton $\mathcal{A}$ satisfies an LTL formula A, in symbols $\mathcal{A} \models A$, is supposed to mean that for any accepting computation sequence s of $\mathcal{A}$ the omega structure ξ (represented as an infinite word over a suitable alphabet) associated with s satisfies the formula, in symbols $\xi \models A$. Defining truth of a formula with respect to an omega structure ξ has the advantage of greater modularity; the theory of the logic LTL can be developed without reference to where omega structures come from. But, when we deal exclusively with the task of model checking, as we will do in Section 5.3, it makes more sense to define directly when an accepting computation sequence $s = (s_i)_{i \geq 0}$ satisfies an LTL formula A. Remember, that $\xi(n)$ is the label of the edge leading from s_n to s_{n+1} and labels are sets of propositional atoms. The definition $s \models A$ now reads:

Definition 134 (Semantics of LTL for computation sequences)

$s \models p$	*iff*	$p \in \xi(0)$ *(p an atom)*
$s \models op(A, B)$		*for propositional connectors* $op(A, B)$ *of* A *und* B *as usual*
$s \models \mathbf{G}A$	*iff*	*for all* $n \in \mathbb{N}$ *it is true that* $s_n \models A$
$s \models \mathbf{F}A$	*iff*	*there is an* $n \in \mathbb{N}$ *satisfying* $s_n \models A$
$s \models A\ \mathbf{U}\ B$	*iff*	*there is* $n \in \mathbb{N}$ *with* $s_n \models B$ *and* $s_m \models A$ *for all* m *with* $0 \leq m < n$
$s \models \mathbf{X}\ A$	*iff*	$s_1 \models A$

Here s_n *is the tail* $(s_m)_{m \geq n}$ *of* s.

Lemma 122 below contains a deep result on the expressive power of LTL formulas. Roughly speaking, LTL formulas can only express periodic properties of computation sequences. The following text makes this statement precise.

Definition 135 (Cyclic Words) *We call an omega word* $w \in V^\omega$ cyclic *if there are finite words* $x, y \in V^*$ *with* $w = xy^\omega$.
In the same way we speak of a cyclic computation sequence s *if* $s = s_0 s_1^\omega$ *is true for finite sequences* s_0, s_1 *of states and of a cyclic omega structure* ξ *if* $\xi = \xi(0) \ldots \xi(i-1)(\xi(i) \ldots \xi(k-1))^\omega$ *for appropriate* i *and* k.

Lemma 122 *Let* ϕ *be an LTL formula using atoms from* PVar.
Let $\mathcal{A}$ *be a Büchi automaton with the vocabulary* $V = \mathcal{P}(\text{PVar})$. *Thus every word* $w \in V^\omega$ *naturally corresponds to an omega structure* ξ_w.
If there is a word $w \in L^\omega(\mathcal{A})$ *with* $\xi_w \models \phi$, *then there is already a cyclic omega word* $xy^\omega \in L^\omega(\mathcal{A})$ *satisfying* $\xi_{xy^\omega} \models \phi$ *and* xy *is an initial segment of* w.

Proof For simple formulas like $\phi \equiv \mathbf{G}p$ or $\phi \equiv \mathbf{F}p$ it can be easily seen that the statement of the lemma is true. If t is an accepting computation sequence for the omega word w with $\xi_w \models \mathbf{G}p$. Then there is a final state s_F that occurs infinitely often in t. Thus there is an initial segment $t_0 \ldots t_{i-1} s_F t_{i+1} \ldots t_k s_F$ of t. The cyclic sequence $t_0 \ldots t_{i-1}(s_F t_{i+1} \ldots t_k)^\omega$ is also an accepting computation sequence of $\mathcal{A}$ and also $w_0 \ldots w_{i-1}(w_i w_{i+1} \ldots w_k)^\omega \models \mathbf{G}p$.

In case $\xi_w \models \mathbf{F}p$ is true, we pick the smallest i such that $p \in w_i$. If t_i occurs before the first occurence of s_F, we proceed as just described, otherwise we

consider the initial subsequence of t up to the first appearance of s_F after t_i. It is however hard to see, how this easy construction can be generalized or replaced by an inductive argument to work for arbitrary ϕ. We will therefore resort to the following trick.

By Theorem 120 there exists a Büchi automaton $\mathcal{B}_\phi$ such that $L^\omega(\mathcal{B}_\phi) = \{\xi \mid \xi \models \phi\}$. Furthermore let $\mathcal{C}$ be the automaton $\mathcal{A} \cap \mathcal{B}_\phi$ from Lemma 121 with the propety $L^\omega(\mathcal{C}) = L^\omega(\mathcal{A}) \cap L^\omega(\mathcal{B}_\phi)$. By the assumption of the lemma we are about to prove we know $L^\omega(\mathcal{C}) \neq \emptyset$. There is thus an accepting computation sequence s' of $\mathcal{C}$, such that the omega structure ξ' associated with s' satisfies $\xi' \models \phi$. By the Büchi acceptance condition there is a final state s_F occuring infinitely often in s'. Let s'_i be the first and s'_k the second occurence of s_F in s' and s the sequence $s'_1, \ldots, s'_{i-1}(s'_i, \ldots, s'_{k-1})^\omega$. Then s also is an accepting computation sequence of $\mathcal{C}$. Consequently the omega structure ξ associated with s satisfies again $\xi \models \phi$.

Now, unfortunately, we have to take a closer look at the internal structure of $\mathcal{C}$. Every state s_j in the sequence s is of the form $s_j = (s^1_j, s^2_j, k_j)$ with s^1_j a state of $\mathcal{A}$, s^2_j a state of $\mathcal{B}_\phi$ and $k_j \in \{1, 2\}$. By construction of $\mathcal{C}$ we know that s^1_F is a final state of $\mathcal{A}$. Thus the projection of s to its first coordinate, $s^1 = s^1_1, \ldots, s^1_n, \ldots$ is an accepting computation sequence of $\mathcal{A}$. The projection s^2 of s to its second coordinate is likewise an accepting computation sequence of $\mathcal{B}_\phi$. Finally, we observe that the omega structures associated with s^1 and s^2 both coincide with ξ. Thus we have found a cyclic omega structure $\xi = \xi(0) \ldots \xi(i-1)(\xi(i) \ldots \xi(k-1))^\omega$ proving the claim of the lemma.

∎

5.2.1 Expressiveness of Linear Temporal Logic

Definition 136 (Partial Isomorphism) *Let $\mathcal{T}_1 = (\mathbb{N}, <, \xi_1)$, $\mathcal{T}_2 = (\mathbb{N}, <, \xi_2)$ be two omega structures. A partial isomorphism, f from $\mathcal{T}_1$ to $\mathcal{T}_2$ is a function with*

1. *domain $dom(f) \subseteq \mathbb{N}$ and range $ran(f) \subseteq \mathbb{N}$*
2. *f is strict order preserving, i.e., $f(n) < f(m) \Leftrightarrow n < m$*
3. *$\xi_1(m) = \xi_2(f(m))$ for all $m \in dom(f)$.*

Note that item 2 entails injectivity, $f(n) = f(m) \Rightarrow n = m$.

We write $f : \mathcal{T}_1 \rightarrowtail \mathcal{T}_2$ to denote that f is a partial isomorphism from $\mathcal{T}_1$ to $\mathcal{T}_2$.

Definition 137 (Partial n-Isomorphism) *The notion of a partial n-isomorphism between two omega structures $\mathcal{T}_1 = (\mathbb{N}, <, \xi_1)$, $\mathcal{T}_2 = (\mathbb{N}, <, \xi_2)$, in symbols $f : \mathcal{T}_1 \overset{n}{\rightarrowtail} \mathcal{T}_2$, is inductively defined.*

1. *$f : \mathcal{T}_1 \overset{0}{\rightarrowtail} \mathcal{T}_2$ iff $f : \mathcal{T}_1 \rightarrowtail \mathcal{T}_2$*
2. *$f : \mathcal{T}_1 \overset{n+1}{\rightarrowtail} \mathcal{T}_2$ iff*
   (a) *for every $m_1 \in \mathbb{N}$ there is $m_2 \in \mathbb{N}$ such that $f' : \mathcal{T}_1 \overset{n}{\rightarrowtail} \mathcal{T}_2$ where f' is the mapping with $dom(f') = dom(f) \cup \{m_1\}$ extending f (i.e., $f(m) = f'(m)$ for $m \in dom(f)$, and $f'(m_1) = m_2$*
   (b) *for every $m_2 \in \mathbb{N}$ there is $m_1 \in \mathbb{N}$ such that $f* : \mathcal{T}_1 \overset{n}{\rightarrowtail} \mathcal{T}_2$ where f^* is the mapping with $dom(f^*) = dom(f) \cup \{m_1\}$ extending f (i.e., $f(m) = f^*(m)$ for $m \in dom(f)$, and $f^*(m_1) = m_2$*

We will write $f' = f \cup \{(m_1, m_2)\}$ and $f^* = f \cup \{(m_1, m_2)\}$.

We observe that $f : \mathcal{T}_1 \overset{n}{\rightarrowtail} \mathcal{T}_2$ implies $f : \mathcal{T}_1 \overset{k}{\rightarrowtail} \mathcal{T}_2$ for all $k \leq n$.

Theorem 123 *Let $\mathcal{T}_1 = (\mathbb{N}, <, \xi^1)$, $\mathcal{T}_2 = (\mathbb{N}, <, \xi^2)$ be two omega structures and f a partial n-isomorphism, $f : \mathcal{T}_1 \overset{2n}{\rightarrowtail} \mathcal{T}_2$.*

For every LTL-formula A with $od(A) \leq n$ we have for all $m \in dom(f)$

$$\xi^1_m \models A \Leftrightarrow \xi^2_{f(m)} \models A$$

Proof Since a formula A with $od(A) = 0$ contains no temporal operator we get $\xi^1_m \models A \Leftrightarrow \xi^2_{f(m)} \models A$ for every $m \in dom(f)$ from the property of f being a partial isomorphism.

So let us assume the claim is true for n and set out to prove it for $n+1$. So, we work from the assumption $f : \mathcal{T}_1 \overset{2n+2}{\rightarrowtail} \mathcal{T}_2$. Now, A is a formula with $od(A) = n+1$. We may restrict attention to the cases $A = \mathbf{X}\ B$ and $A = B_1\ \mathbf{U}\ B_2$.

Assume $\xi^1_m \models \mathbf{X}\ B$. Thus $\xi^1_{m+1} \models B$. By definition of an $2n+2$ isomorphism there is k such that $f' : \mathcal{T}_1 \overset{2n+1}{\rightarrowtail} \mathcal{T}_2$. for $f' = f \cup \{(m+1, k)\}$. Since f' is strictly order preserving we know $f(m) < k$. We claim that even $k = f(m) + 1$ is true. Otherwise we would find m_1 with $f(m) < m_1 < k$. But then there has to be some m_0 such that $f'' : \mathcal{T}_1 \overset{2n}{\rightarrowtail} \mathcal{T}_2$ for $f'' = f' \cup \{(m_0, m_1)\}$. This would imply $m < m_0 < m+1$, a contradiction. Thus $f'(m+1) = f(m) + 1$. From $f' : \mathcal{T}_1 \overset{2n}{\rightarrowtail} \mathcal{T}_2$ and the induction hypothesis we obtain $\xi^2_{f'(m+1)} \models B$, thus $\xi^2_{f(m)+1} \models B$, thus $\xi^2_{f(m)} \models \mathbf{X}\ B$. The reverse implication $\xi^2_{f(m)} \models \mathbf{X}\ B \Rightarrow \xi^1_m \models \mathbf{X}\ B$ is proved analogously.

Next assume $\xi^1_m \models B_1\ \mathbf{U}\ B_2$. Thus there is m_0 such that

1. $m \leq m_0$
2. $\xi^1_{m_0} \models B_2$
3. $\xi^1_{m'} \models B_1$ for all $m \leq m' < m_0$.

In case $m_0 = m$ the induction hypothesis yields $\xi^2_{f(m)} \models B_2$ and thus $\xi^2_{f(m)} \models B_1\ \mathbf{U}\ B_2$. So we assume $m < m_0$ from now on. By definition of $2n+2$ partial isomorphisms there is k such that $f' : \mathcal{T}_1 \overset{2n+1}{\rightarrowtail} \mathcal{T}_2$ for $f' = f \cup \{(m_0, k)\}$. By induction hypothesis this gives already $\xi^2_k \models B_2$. Assume there is k', $f(m) \leq k'_1 < k$ with $\xi^2_{k'_1} \models \neg B_1$. There would then be some k'_0 with $f'' : \mathcal{T}_1 \overset{2n}{\rightarrowtail} \mathcal{T}_2$ for $f'' = f' \cup \{(k'_0, k'_1)\}$. Since f'' is strictly order preserving we must have $m \leq k'_0 < m_0$. Since $od(\neg B_1) = od(B_1)$ we obtain furthermore, by induction hypothesis $\xi^1_{k'_0} \models \neg B_1$. A contradiction to the choice of m_0. This proves alltogether $\xi^2_{f(m)} \models B_1\ \mathbf{U}\ B_2$. The reverse implication $\xi^2_{f(m)} \models B_1\ \mathbf{U}\ B_2 \Rightarrow \xi^1_m \models B_1\ \mathbf{U}\ B_2$ is again proved analogously.

∎

5.3 Bounded Model Checking (Optional)

In this section we will present an interesting connection between LTL model checking and propositional satisfiability. It is the theoretical basis for an alternative to model checking with Büchi automata by employing programs to solve propositional satisfiability problem, commonly called SAT solvers. This approach goes by the name *bounded model checking* and was pioneered in the paper [Biere *et al.*, 1999].

Our starting point are statements of the form $s \models A$, the LTL formula A is satiesfied for the computation sequence s, as detailed in Definition 134.

In the course of this section we will express statements of the form, $s \models A$, by a propositional formula. The first problem with this plan is the infinity of s. Its solution has already been prepared by Lemma 122: it suffices to consider finite, cyclic computation sequences.

We are now ready to start with propositional coding.

Lemma 124 *For any given Büchi automaton $\mathcal{A}$, any LTL formula F and any $k \in \mathbb{N}$ there is a propositional formula M_k such that*

$$M_k \text{ is satisfiable} \quad \text{iff} \quad \text{there is a cyclic computation sequence } s \text{ of length } k \text{ satisfying } s \models F$$

If necessary we will write more precisely $M_k(\mathcal{A}, F)$ instead of M_k.

Proof We will describe an explicit construction for M_k. This construction is quite involved, but easy to follow. We will therefore not include an additional proof that the proposed construction really does what it should. We will illustrate the different steps for the special case of the automaton $\mathcal{A}_{dbp}$ from Figure 5.2 and the formula $F = \mathbf{FG}p$.

Let us assume that $\mathcal{A}$ has n states. We will refer to these states by using the numbers from 1 to n. As a first preparatory step we need a binary coding of these state numbers. The coding will use the Boolean variables $c_1, \ldots, c_m$.

The example automaton $\mathcal{A}_{dbp}$ uses only two states, thus one Boolean variable c suffices. $I(c) = \mathbf{false}$ characterizes the initial state and $I(c) = \mathbf{true}$ the uniques final state.

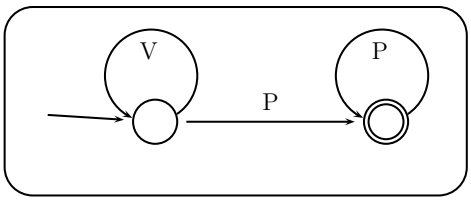


Figure 5.2: The example automaton $\mathcal{A}_{dbp}$

The formula F also contains propositional variables. Let these be $\Sigma = \{p_1, \ldots, p_r\}$.

In the example p and q are the only propositional variables in Σ even though only p occurs in our example formula.

Later on we will need additional auxiliary variables which we will introduce when we get there.

Since the formula M_k will have to talk about sequences of states of length k there will be k copies of each variable, i.e., c_j^i with $1 \leq i \leq k$ und $1 \leq j \leq m$ and p_j^i with $1 \leq i < k$ und $1 \leq j \leq r$.

In our example we will work with $k = 3$. Thus we have the propositional variables c^1, c^2, c^3, p^1, p^2 and q^1, q^2 at our disposal.

The formula M_k is made up of several parts

$$M_k \equiv Init \wedge Trans \wedge \bigvee_{1 \leq i < k} L_i$$

For any interpration I of the propositional variables we read the values $I(c_1^i), \ldots, I(c_m^i)$ for every $1 \leq i \leq k$ as the binary encoding of a number n_i. This gives us a sequence of states $\pi = n_1, \ldots, n_k$. Likewise we read $I(p_1^i), \ldots, I(p_r^i)$ as the coding of a subset b_i of the variables $p_1, \ldots, p_r$. Thus b_i is a letter in the vocabulary V of edge labels of the automaton $\mathcal{A}$. Taken all together the b_i form a word $b_1, \ldots, b_{k-1}$ of length $k-1$. The formulas $Init$ and $Trans$ will take care, when satisfied by I, that $n_1, \ldots, n_k$ is a legal computation sequence of $\mathcal{A}$ with $b_1, \ldots, b_{k-1}$ as its sequence of edge labels.

Let us consider in our example the following interpretations I_1, I_2

	c^1	c^2	c^3	p^1	p^2	q^1	q^2
I_1	0	1	1	1	1	1	0
I_2	0	1	1	1	0	0	1

I_1 and I_2 code the sequence $\pi = 0, 1, 1$ of states. I_1 codes the sequence of letters $w_1 = \{p, q\}, \{p\}$ and I_2 the word $w_2 = \{p\}, \{q\}$. Obviously w_1 is a correct edge labeling for the state sequence π, while w_2 is not.

If $d = d_1, \ldots, d_m$ is the binary code for a state of $\mathcal{A}$ then $S^i_d = S^i_{d_1,\ldots,d_m}$ is used to denote the propositional formula $\bigwedge_{d_j=1} c^i_j \wedge \bigwedge_{d_j=0} \neg c^i_j$. If b is a letter of the alphabet V, that is to say $b \subseteq \Sigma$, then B^i_b denotes the formula $\bigwedge_{p_j \in b} p^i_j \wedge \bigwedge_{p_j \notin b} \neg p^i_j$.
If $d_0 = d_1, \ldots, d_m$ is the binary code of the inital state of $\mathcal{A}$ then

$$Init \equiv S^1_{d_0}$$

Let $(d^1_a, d^1_e, b^1), \ldots (d^K_a, d^K_e, b^K)$ be all edges of the automaton $\mathcal{A}$. An edge is seen here as a transition form a state with binary code d^j_a to a state with binary code d^j_e and edge label b^j for $1 \leq j \leq K$.

$$Trans \equiv \bigwedge_{1 \leq i < k} \bigvee_{1 \leq j \leq K} (S^i_{d^j_a} \wedge S^{i+1}_{d^j_e} \wedge B^i_{b^j})$$

For our example we get $Init = \neg c^1$ and the set of all edges is

$$\begin{array}{l}(0,0,\{\}), (0,0,\{p\}), (0,0,\{q\}), (0,0,\{p,q\}) \\ (0,1,\{p\}), (0,1,\{p,q\}) \\ (1,1,\{p\}), (1,1,\{p,q\})\end{array}$$

which yields

$$\begin{array}{rl} & (\neg c^1 \wedge \neg c^2) \vee (\neg c^1 \wedge c^2 \wedge p^1) \vee (c^1 \wedge c^2 \wedge p^1) \\ Trans = & \wedge \\ & (\neg c^2 \wedge \neg c^3) \vee (\neg c^2 \wedge c^3 \wedge p^2) \vee (c^2 \wedge c^3 \wedge p^2) \end{array}$$

To obtain this simple formula we have already performed some simplifications, e.g., $(\neg c^1 \wedge \neg c^2 \wedge p^1 \wedge q^1) \vee (\neg c^1 \wedge \neg c^2 \wedge \neg p^1 \wedge q^1) \vee (\neg c^1 \wedge \neg c^2 \wedge p^1 \wedge \neg q^1) \vee (\neg c^1 \wedge \neg c^2 \wedge \neg p^1 \wedge \neg q^1)$ has been equivalently replaced by $(\neg c^1 \wedge \neg c^2)$. As an additional simplification $(c^1 \wedge c^2 \wedge p^1)$ could be dropped since it contradicts $Init$.

The formulas L_i will guarantee that the finite computation sequence π extracted from a satisfying interpretation I is i-cyclic and accepting and, most importantly , satisfies the formla F:

$$L_i = Z_i \wedge Akz_i \wedge erfF_i$$

The first two components are easy

$$\begin{array}{lcl} Z_i & \equiv & \wedge_{1\leq j\leq m}(c_j^k \leftrightarrow c_j^i) \\ Akz_i & \equiv & Fin_i \vee Fin_{i+1} \ldots \vee Fin_{k-1} \end{array}$$

with $Fin_i \equiv S^i_{d_1^f} \vee \ldots \vee S^i_{d_R^f}$ where $d_1^f, \ldots, d_R^f$ are the binary codes of all final states of $\mathcal{A}$.

It remains to explain $erfF_i$. We want that satisfiability of this formula by an interpretation I guarantees that the i-cyclic computation sequence π extracted from I satisfies formula F. At this point we need additional propositional variables. It is true that we are only interested in validity or non-validity of F at position 1 in π, but by the definition of the temporal operators it cannot be avoided to consider validity at all positions j with $1 \leq j \leq k$. In addition validity depends on the cyclic loop-back index i. We need thus for every subformula C, for every i, $1 \leq i < k$ and every j, $1 \leq j \leq k$ a new propositional variable denoted by $[C]_i^j$. The defining formulas for every $[C]_i^j$ are listed in Figure 5.3. The entries for $\mathbf{G}C$ and $\mathbf{F}C$ are infact superfluous. But, having seen the definitions of the simple formulas $\mathbf{G}C$ and $\mathbf{F}C$ may help to understand the case of the more complicated operators $C_1\ \mathbf{U}\ C_2$ and $C_1\ \mathbf{V}\ C_2$. There is no entry involving the negation symbol. This is because we assume that F is given in negation normal form. In particular the formulas $\neg C_2$ on the right hand side of the definition for $[C_1\ \mathbf{V}\ C_2]_i^j$ are understood as a short hand notation for its negation normal form. In addition to the propositional equivalences in Figure 5.3 we need the final equivalence

$$erfF_i \leftrightarrow [F]_i^1$$

Let us now compute the formulas L_1 and L_2 for our running example. We start with $Z_1 \equiv c^1 \leftrightarrow c^3$ and $Z_2 \equiv c^2 \leftrightarrow c^3$. Since c is the only final state of $\mathcal{A}_{dbp}$ we get $Akz_1 \equiv c^1 \vee c^2 \vee c^3$ and $Akz_2 \equiv c^2 \vee c^3$. For our example formula $F \equiv \mathbf{FG}p$ we obtain the following instances of the equivalences from Figure 5.3.

$$\begin{array}{lcl} [F]_1^1 & \equiv & [\mathbf{G}p]_1^1 \vee [\mathbf{G}p]_1^2 \\ [\mathbf{G}p]_1^1 & \equiv & [p]_1^1 \wedge [p]_1^2 \\ & \equiv & p^1 \wedge p^2 \\ [\mathbf{G}p]_1^2 & \equiv & [p]_1^2 \wedge [p]_1^1 \\ & \equiv & p^2 \wedge p^1 \end{array}$$

$$
\begin{array}{lcll}
[C]_i^j & \leftrightarrow & p_l^j \quad \text{if } C = p_l \in \Sigma & \\
[C]_i^j & \leftrightarrow & \neg p_l^j \quad \text{if } C = \neg p_l \text{ with } p_l \in \Sigma & \\
[C_1 \wedge C_2]_i^j & \leftrightarrow & [C_1]_i^j \wedge [C_2]_i^j & \\
[C_1 \vee C_2]_i^j & \leftrightarrow & [C_1]_i^j \vee [C_2]_i^j & \\
[\mathbf{G}C]_i^j & \leftrightarrow & \bigwedge_{j \le l < k} [C]_i^l & \text{if } j \le i \\
[\mathbf{G}C]_i^j & \leftrightarrow & \bigwedge_{j \le l < k} [C]_i^l \wedge \bigwedge_{i \le l < j} [C]_i^l & \text{if } i < j \\
[\mathbf{F}C]_i^j & \leftrightarrow & \bigvee_{j \le l < k} [C]_i^l & \text{if } j \le i \\
[\mathbf{F}C]_i^j & \leftrightarrow & \bigvee_{j \le l < k} [C]_i^l \vee \bigvee_{i \le l < j} [C]_i^l & \text{if } i < j \\
[C_1 \ \mathbf{U} \ C_2]_i^j & \leftrightarrow & \bigvee_{j \le l < k} ([C_2]_i^l \wedge \bigwedge_{j \le n < l} [C_1]_i^n) & \text{if } j \le i \\
[C_1 \ \mathbf{U} \ C_2]_i^j & \leftrightarrow & \bigvee_{j \le l < k} ([C_2]_i^l \wedge \bigwedge_{j \le n < l} [C_1]_i^n) \vee & \\
 & & \bigvee_{i \le l < j} ([C_2]_i^l \wedge \bigwedge_{j \le n < k} [C_1]_i^n \wedge \bigwedge_{i \le n < l} [C_1]_i^n) & \text{if } i < j \\
[C_1 \ \mathbf{V} \ C_2]_i^j & \leftrightarrow & [C_2]_i^j \wedge \bigwedge_{i \le l < k} ([\neg C_2]_i^l \rightarrow \bigvee_{i \le n < l} [C_1]_i^n) & \text{if } j \le i \\
[C_1 \ \mathbf{V} \ C_2]_i^j & \leftrightarrow & [C_2]_i^j \wedge \bigwedge_{j \le l < k} ([\neg C_2]_i^l \rightarrow \bigvee_{j \le n < l} [C_1]_i^n) \wedge & \\
 & & \bigwedge_{i \le l < j} ([\neg C_2]_i^l \rightarrow \bigvee_{i \le n < l} [C_1]_i^n \vee \bigvee_{j \le n < k} [C_1]_i^n) & \text{if } i < j \\
[\mathbf{X} \ C]_i^j & \leftrightarrow & [C]_i^{j+1} & \text{if } j < (k-1) \\
[\mathbf{X} \ C]_i^{k-1} & \leftrightarrow & [C]_i^i &
\end{array}
$$

Figure 5.3: Cyclic Semantics for LTL formulas

In total this amounts to $[F]_1^1 \leftrightarrow p^1 \wedge p^2$

$$
\begin{array}{lcl}
[F]_2^1 & \equiv & [\mathbf{G}p]_2^1 \vee [\mathbf{G}p]_2^2 \\
[\mathbf{G}p]_2^1 & \equiv & [p]_2^1 \wedge [p]_2^2 \\
 & \equiv & p^1 \wedge p^2 \\
[\mathbf{G}p]_2^2 & \equiv & [p]_2^2 \\
 & \equiv & p^2
\end{array}
$$

This can be summarized to $[F]_2^1 \leftrightarrow p^2$.

The formulas L_i for our example can now be computed as:

$$
\begin{array}{lcl}
L_1 & \leftrightarrow & (c^1 \leftrightarrow c^3) \wedge (c^1 \vee c^2 \vee c^3) \wedge p^1 \wedge p^2 \\
 & \leftrightarrow & (c^1 \leftrightarrow c^3) \wedge (c^2 \vee c^3) \wedge p^1 \wedge p^2 \\
L_2 & \leftrightarrow & (c^2 \leftrightarrow c^3) \wedge (c^2 \vee c^3) \wedge p^2 \\
 & \leftrightarrow & c^2 \wedge c^3 \wedge p^2
\end{array}
$$

The result of the computation i.e., M_3 is presented in Figure 5.4. This formula can be further simplified making use of the equation $Init = \neg c^1$.

This leads to $c^2 \rightarrow p^1$ which may in turn be used for further simplification. This leads to the final result $M_3 \equiv \neg c^1 \wedge c^2 \wedge c^3 \wedge p^1 \wedge p^2$. This formula is obviously satisfiable. By construction this tells us that there is a computation sequence s in $\mathcal{A}_{dbp}$ that satisfies $\mathbf{FG}p$. We even know that this formula is true for every computation sequence of $\mathcal{A}_{dbp}$.

$$
\begin{array}{l}
\neg c^1 \wedge \\
(\neg c^1 \wedge \neg c^2) \vee (\neg c^1 \wedge c^2 \wedge p^1) \vee (c^1 \wedge c^2 \wedge p^1) \\
\wedge \\
(\neg c^2 \wedge \neg c^3) \vee (\neg c^2 \wedge c^3 \wedge p^2) \vee (c^2 \wedge c^3 \wedge p^2) \\
\wedge \quad (\\
((c^1 \leftrightarrow c^3) \wedge (c^2 \vee c^3) \wedge p^1 \wedge p^2) \\
\vee \\
(c^2 \wedge c^3 \wedge p^2) \\
\qquad)
\end{array}
$$

Figure 5.4: M_3 for the example automaton $\mathcal{A}_{dbp}$ and $F \equiv \mathbf{FG}p$

Theorem 125 *There is an accepting computation sequence of a Büchi automaton $\mathcal{A}$ satisfying the LTL-formula F iff there exists k, such that the propositional formula $M_k(\mathcal{A}, F)$ is satisfiable.*

Proof Follows from Lemmata 124 and 122.

∎

The bounded model checking procedure for a Büchi automaton $\mathcal{A}$ and an LTL formula F now works as follows: Analyze satifiability of the set $M_k(\mathcal{A}, F)$ first for small values of k. This can be done by using powerful implementations of propositional decision procedures, called *SAT solver*. If a solution is found we have also positively solved the original LTL problem. If on the other hand $M_k(\mathcal{A}, F)$ is found to be not satisfiable on can continue to analyze satisfiability of $M_{k+1}(\mathcal{A}, F)$. In case there is no computation sequence of $\mathcal{A}$ satisfying F this procedure will not terminate. But, it is not hard to see that the proof of Lemma 122 also gives an upper bound for k. The least upper bound such that Lemma 122 is true is called the (*completeness threshhold*). If $M_k(\mathcal{A}, F)$ is not satisfiable for some k greater or equal to the completeness threshhold then we know that no accepting computation sequence of $\mathcal{A}$ satisfying F exists.

Bounded model checking does no reduce the computational complexity of the problem to solve. This is still based on an NP-complete problem. Experiments have shown however that the advantages of the automata based approach to LTL model checking are in a sense orthogonal to the advantages of bounded model checking, [Biere *et al.*, 2003, Copty *et al.*, 2001].

5.4 Computation Tree Logic

In this section we will present and study *Computation Tree Logic* abbreviated as *CTL* as a typical representative of what are called branching time temporal logics. Among other reference our account is based on [Clarke *et al.*, 1986, Emerson, 1992, Clarke *et al.*, 2001] and [Huth & Ryan, 2000]. In particular the example in the following subsection is taken from [Huth & Ryan, 2000].

Motivating Example

To convey a first idea of the kind of problems that can be formulated and solved using Computation Tree Logic we reconsider an important and popular topic in Computer Science: mutual exclusion. In this scenario a number of actors compete for a common resource that can only be used by one party at the time. You may think of several threads of a concurrent program to read from and write to the same file. For the sake of simplicity we will in this subsection only consider two actors. The principal issues can already be observed in this simplest case. We use a rather abstract approach and model the given scenario by a transition system. A more concrete treatment can be found in [Lamport, 1974]. In our abstract model an agent can either be not active, trying to obtain the critical resource or be in the critical section. This will be modeled by the six boolean variables n_i, t_i and c_i for $i \in \{1, 2\}$. The transition system (S, s_0, R, v) consists of a set S of system states with an initial state s_0, of a binary relation R stating which transitions are possible among the states in S and an evaluation function $v : S \times \mathit{atoms} \to \{\mathbf{true}, \mathbf{false}\}$ that states which boolean atoms are true in which states. Figure 5.5 shows a graphical representation of the transition system $A_1 = (S_1, s_0, R_1, v_1)$ that we will consider first.

The node labeling $t_1 n_2$ in state s_1 e.g., means that in the global system state s_1 the first agent is in the trying phase while the second agent is not active. Using the terminology from the transition system R_1 this can be expressed as $v_1(s_1, t_1) = \mathbf{true}$ and $v_1(s_1, n_2) = \mathbf{true}$ or more concise as $(R_1, s_1) \models t_1$ and $(R_1, s_1) \models n_2$. Since we are talking about a concurrent system it is not determined what will be the next system state after s_1. If agent 1 will be served next we end up in state s_2, if agent 2 will be served next we end up in state s_3. This is reflected in the non-deterministic transition system. In some states there is no choice, e.g., in state s_4 agent 2 has to wait till agent

1 leaves the critical section. What kind of properties do we want to assert about R_1. Here are five examples

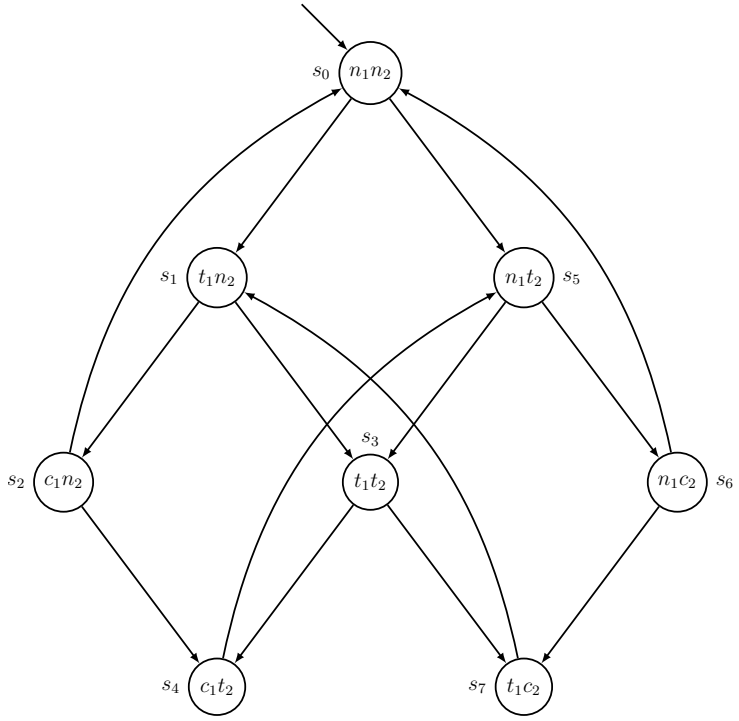


Figure 5.5: Mutual Exclusion (first attempt)

safety There is no state s reachable from S_0 with $s \models c_1 \wedge c_2$.

liveness Whenever an agent wants to enter the critical section it will eventually enter it.

non-blocking An agent can always try to enter the critical section.

non-sequencing It is not the case that the agent who first tried will first enter the critical section.

non-alternating It is not the case that the two agents take alternate turns to the critical section.

The safety property is obvious. The state with c_1c_2 is not even included in Figure 5.5 since it has no incoming edge. The non-blocking property can also seen to be true. It may also be observed that there are no dead ends. An example of the possible behaviour required by non-sequencing is the sequence of states s_0, s_1, s_3, s_7. The sequence s_0, s_1, s_2, s_0 is an example of a behaviour where the first agent enters the critical section two times in a row. We have so far avoided to consider the liveness property, which in fact is not satisfied. In the sequence $s_0, s_1, s_3, s_7, s_1, s_3, s_7, \ldots$ agent 1 never reaches the critical section. In the second attempt to model mutual exclusion $R_2 = (S_2, R_2, v_2)$ depicted in Figure 5.6 liveness is guaranteed. But now the non-sequencing

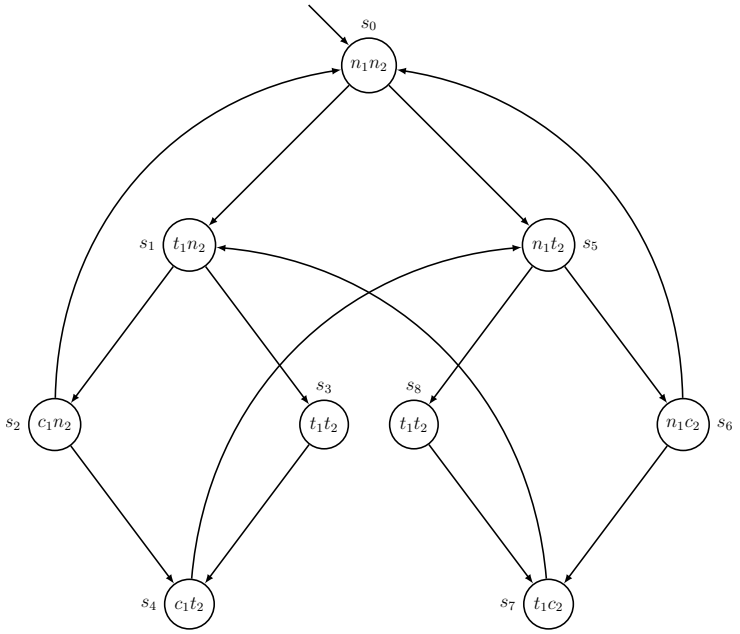


Figure 5.6: Mutual Exclusion (second attempt)

property is violated. In the next subsection we will present a formal language that allows to express the five properties considered above, and many more. In further subsections to follow we will learn methods that automate the evaluation wether a given property is true in a given transition system.

Definition 138 (Transition System)
Let PVar *be a set of propostional atoms.*
A transition system $\mathcal{T} = (S, s_0, R, v)$ *consists of*

- *a finite set S of states with one distinguished initial state s_0,*
- *a binary relation R and*
- *a function $v : S \times \mathrm{PVar} \to \{\mathbf{true}, \mathbf{false}\}$*

such that for every $s \in S$ there is $s' \in S$ with $R(s, s')$.
If necessary we call $\mathcal{T}$ a transition system over PVar *or say that* PVar *are the atoms for $\mathcal{T}$.*

From a technical point of view a transition system is just a Kripke structure, see Definition 37 on page 75, whose accessability relation has no dead ends. Also Kripke structures do not come equipped with an initial state. It will turn out, that the initial state of a transition system usually does not play a role and we will most of the time write (S, R, v) instead of (S, s_0, R, v). The similarity to Kripke structures is at least true for the simple version of transition systems that we consider here. There are more complicated versions e.g., labeled transitions systems that cannot, at least not directly, be mimicked by Kripke structures. More important is the question where transitions come from.

Syntax and Semantics of CTL

Like all temporal logics we consider in this text also CTL is a propositional logic. There is thus a set PVar of propositional variables to start with.

Definition 139 (Syntax of CTL)

1. *Any propositional variable $p \in$ PVar is a CTL formula.*
2. *If F, G are CTL formulas then all propositional combinations are also CTL formulas, e.g., $\neg F$, $F \vee G$, $F \wedge G$, etc.*
3. *If F, G are CTL formulas then also*

$$\mathbf{AX}F, \mathbf{EX}F, \mathbf{A}(F\ \mathbf{U}\ G) \text{ and } \mathbf{E}(F\ \mathbf{U}\ G)$$

are CTL formulas.

If p, q are atoms in PVar then $\mathbf{AX}(p \to q)$, $\mathbf{A}(p\,\mathbf{U}\,q)$ and $\mathbf{A}(\mathbf{E}(p\,\mathbf{U}\,q)\,\mathbf{U}\,G\neg q)$ are CTL formulas, while $p\ \mathbf{U}\ q$ and $\mathbf{A}p$ are not syntactically correct. The operators always come in pairs. There can be no $\mathbf{U}$ or $\mathbf{X}$ without $\mathbf{A}$ or $\mathbf{E}$ preceeding it and no $\mathbf{A}$ or $\mathbf{E}$ without an $\mathbf{X}$ or $\mathbf{U}$ following it. This may seem a bit strange at the moment. After we have seen the temporal logic CTL* in the next subsection the definition of CTL will look more natural.

The meaning of a CTL formula will be defined relative to a given transition system (S, R, v). The crucial concept will be that of a path through (S, R, v).

Definition 140 (Path)
A path through a transition system $\mathcal{T} = (S, R, v)$ is an infinite sequence of states $t(0), t(1), \ldots, t(n), t(n+1), \ldots$ such that $t(0)$ is the initial state and for all n the relation $R(t(n), t(n+1))$ is true.
If π denotes a path then $\pi(n)$ is the state at position n and π_n is the tail of π starting at n, i.e.,

$$\pi_n(k) = \pi(n+k).$$

We do not *insists that a path of $\mathcal{T}$ starts with the initial state.*

For the transition system in Figure 5.6 the sequence $s_0, s_1, s_2, s_0, s_1, s_2, \ldots$ is an example of a path. There is the possibility that the fact that the same state may occur at different positions within a path may lead to confusion. Experience with other texts shows that the possibility is rather slight. So we will live with it. The alternative would have been to consider a path as a sequence of abstract time points, as we did for LTL. For the moment this is not a matter of high priority.

Definition 141 (Semantics of CTL) *Let $\mathcal{T} = (S, R, v)$ be a transition system. We will define when a CTL formula ϕ is true in a state s of $\mathcal{T}$. As usual this will be symbolically denoted by $(\mathcal{T}, s) \models \phi$. Here and in all cases where no ambiguity can arise we will write $s \models \phi$ instead of $(\mathcal{T}, s) \models \phi$.*

1	$s \models p$	*iff*	$v(s,p) = 1$ (*in case* $p \in \mathrm{PVar}$)
2	$s \models \neg\phi$	*iff*	$s \not\models \phi$
3	$s \models \phi_1 \wedge \phi_2$	*iff*	$s \models \phi_1$ *and* $s \models \phi_2$
4	$s \models \mathbf{AX}\phi$	*iff*	$s_1 \models \phi$ *is true for all* s_1 *with* $R(s,s_1)$
5	$s \models \mathbf{EX}\phi$	*iff*	$s_1 \models \phi$ *is true for at least one* s_1 *with* $R(s,s_1)$
6	$s \models \mathbf{A}(\phi_1\ \mathbf{U}\ \phi_2)$	*iff*	*for every path* $s_0, s_1, \ldots$ *with* $s_0 = s$ *there exists* $i \geq 0$, *such that* $s_i \models \phi_2$ *and* $s_j \models \phi_1$ *for all* j *with* $0 \leq j < i$,
7	$s \models \mathbf{E}(\phi_1\ \mathbf{U}\ \phi_2)$	*iff*	*there is a path* $s_0, s_1, \ldots$ *with* $s_0 = s$ *and there is* $i \geq 0$, *such that* $s_i \models \phi_2$ *and* $s_j \models \phi_1$ *for all* j *satisfying* $0 \leq j < i$,

Definition 142 (Defined CTL Operators)
Using the operators **F** *and* **G** *from LTL (see Lemma 118 on page 223) four new CTL operators can be defined:*

$ua(\phi)$	$\equiv$	$\mathbf{AF}\phi$	$\equiv$	$\mathbf{A}(\mathbf{true}\ \mathbf{U}\ \phi)$	ϕ *cannot be avoided*
$re(\phi)$	$\equiv$	$\mathbf{EF}\phi$	$\equiv$	$\mathbf{E}(\mathbf{true}\ \mathbf{U}\ \phi)$	ϕ *is reachable*
$ofa(\phi)$	$\equiv$	$\mathbf{EG}\phi$	$\equiv$	$\neg\mathbf{A}(\mathbf{true}\ \mathbf{U}\ \neg\phi)$	*once and for all* ϕ
$aw(\phi)$	$\equiv$	$\mathbf{AG}\phi$	$\equiv$	$\neg\mathbf{E}(\mathbf{true}\ \mathbf{U}\ \neg\phi)$	*always* ϕ

To work with these defined temporal operators it will be helpful to state their semantics in the style of Definition 141:

Lemma 126 (Semantics of Defined Operators)

8	$s \models \mathbf{AF}\phi$	*iff*	*for every path* $s_0, s_1, \ldots$ *with* $s_0 = s$ *there exists* $i \geq 0$, *such that* $s_i \models \phi$
9	$s \models \mathbf{EF}\phi$	*iff*	*there is a path* $s_0, s_1, \ldots$ *with* $s_0 = s$ *and there exists* $i \geq 0$, *such that* $s_i \models \phi$
10	$s \models \mathbf{EG}\phi$	*iff*	*there is a path* $s_0, s_1, \ldots$ *with* $s_0 = s$ *such that* $s_i \models \phi$ *for all* i
11	$s \models \mathbf{AG}\phi$	*iff*	*for every path* $s_0, s_1, \ldots$ *with* $s_0 = s$ *and every* i *it is true that* $s_i \models \phi$

Proofs Easy consequences from Definitions 142 and 141.

▪

Example 36 *The five properties from Subsection 5.4 on page 238 can be formaly expressed as CTL formulas. We still assume here that there are just two participants.*

safety $s_0 \models \mathbf{AG}\neg(c_1 \wedge c_2)$

liveness $s_0 \models \mathbf{AG} \bigwedge_{i\in\{1,2\}}(t_i \rightarrow \mathbf{EF}c_i)$

non-blocking $s_0 \models \mathbf{AG} \bigwedge_{i\in\{1,2\}}(\mathbf{EF}t_i)$

non-sequencing $\mathbf{EF}(t_1 \wedge \mathbf{EXE}(\neg c_1\ \mathbf{U}\ c_2)) \wedge \mathbf{EF}(t_2 \wedge \mathbf{EXE}(\neg c_2\ \mathbf{U}\ c_1))$

non-alternating $\mathbf{EF}(c_1 \wedge \mathbf{EXE}(\neg c_2\ \mathbf{U}\ c_1)))$

Lemma 127 (CTL Tautologies)
The following formulas are CTL tautologies

1. $\mathbf{AG}\ \phi \leftrightarrow \phi \wedge \mathbf{AXAG}\ \phi$
2. $\mathbf{EG}\ \phi \leftrightarrow \phi \wedge \mathbf{EXEG}\ \phi$
3. $\mathbf{AF}\ \phi \leftrightarrow \phi \vee \mathbf{AXAF}\ \phi$
4. $\mathbf{EF}\ \phi \leftrightarrow \phi \vee \mathbf{EXEF}\ \phi$
5. $\mathbf{A}(\phi\ \mathbf{U}\ \psi) \leftrightarrow \psi \vee (\phi \wedge \mathbf{AXA}(\phi\ \mathbf{U}\ \psi))$
6. $\mathbf{E}(\phi\ \mathbf{U}\ \psi) \leftrightarrow \psi \vee (\phi \wedge \mathbf{EXE}(\phi\ \mathbf{U}\ \psi))$

Proofs Let $\mathcal{T} = (S, R, v)$ be a transition system, $g \in S$.
(1) We need to show $g \models \mathbf{AG}\ \phi$ iff $g \models \phi \wedge \mathbf{AXAG}\ \phi$

$g \models \mathbf{AG}\ \phi$ iff for all path $g_0, g_1, \ldots$ with $g_0 = g$ and all i $g_i \models \phi$
iff $g_0 \models \phi$ and
for all path $h_0, h_1, \ldots$ with $h_0 = g_1$ and all i $h_i \models \phi$
iff $g \models \phi$ and
for all h with $R(g, h)$ $h \models \phi$ and
for all path $h_0, h_1, \ldots$ with $h_0 = h$ and all i $h_i \models \phi$
iff $g \models \phi \wedge \mathbf{AXAG}\ \phi$

The remaining parts are proved similarly.

Syntax and Semantics of CTL*

In the definition of CTL* formulas we distinguish between state formulas and path formulas. A state formula F can be evaluated in every state s, in contrast we need to know a whole path p to determine the truth or falsity of a path formula.

Definition 143 (Syntax of CTL*)

1. *any propositional variable is a state formula*
2. *if F, G are state formulas, so are $\neg F$, $F \vee G$, $F \wedge G$, etc.,*
3. *if F is a path formula, then $(\mathbf{A}F)$, $(\mathbf{E}F)$ are state formulas,*
4. *every state formula also is a path formula,*
5. *if F, G are path formulas, so are $\neg F$, $F \vee G$, $F \wedge G$,*
6. *if F, G are path formulas, so $\mathbf{X}F$ und $F\ \mathbf{U}\ G$.*

Definition 144 (Semantics of CTL*) *Let $\mathcal{T} = (S, R, v)$ be a transition system. For any state $s \in S$ and path π of $\mathcal{T}$ we define for state formulas F, F_i and path formulas P, P_i:*

1	$s \models a$	$\Leftrightarrow$	$v(s,a) = \mathbf{true}$	*if* $F = a$
2	$s \models F_1 \wedge F_2$	$\Leftrightarrow$	$s \models F_1$ *and* $s \models F_2$	$\ldots$
	as usual for all propositional connectives			
3	$s \models \mathbf{A}P_1$	$\Leftrightarrow$	$\pi \models P_1$ *for all paths* π *starting with* s	
	$s \models \mathbf{E}P_1$	$\Leftrightarrow$	$\pi \models P_1$ *for some path* π *starting with* s	
4	$\pi \models F$	$\Leftrightarrow$	$\pi(0) \models F$	
5	$\pi \models P_1 \wedge P_2$	$\Leftrightarrow$	$\pi \models P_1$ *and* $\pi \models P_2$	$\ldots$
	as usual for all propositional connectives			
6	$\pi \models \mathbf{X}F$	$\Leftrightarrow$	$\pi_1 \models F$	
	$\pi \models F\ \mathbf{U}\ G$	$\Leftrightarrow$	*there exists an* $n \geq 0$ *with* $\pi_n \models G$ *and* $\pi_m \models F$ *for all* m *with* $0 \leq m < n$	

Remember that π_n is the tail of π starting at n.
A CTL state (path) formula ϕ is a* tautology *if ϕ is true in all states (all paths) for all transition systems.*

We use **F** and **G** as abbreviations for **true U** A and $\neg$(**true U** $\neg A$).

Lemma 128 (Simple CTL* Tautologies)
Let F be a state formula then the equivalences

$$\mathbf{A}F \leftrightarrow F \qquad \mathbf{E}F \leftrightarrow F$$

are tautologies.

Proofs: We fix an arbitrary transition system $\mathcal{T} = (S, R, v)$. References are to clauses of Def. 144

$s \models \mathbf{A}F$	$\Leftrightarrow$	$\pi \models F$ for all path π with $\pi(0) = s$	clause 3
	$\Leftrightarrow$	$\pi(0) \models F$	clause 4
	$\Leftrightarrow$	$\pi \models F$ for some path π with $\pi(0) = s$	$\mathcal{T}$ has no dead ends
	$\Leftrightarrow$	$s \models \mathbf{E}F$	clause 3

∎

Corollary 129 *Let P be a path formula then the following equivalences are tautologies:*

1. $\mathbf{AE}P \leftrightarrow \mathbf{E}P$
2. $\mathbf{EE}P \leftrightarrow \mathbf{E}P$
3. $\mathbf{AA}P \leftrightarrow \mathbf{A}P$
4. $\mathbf{EA}P \leftrightarrow \mathbf{A}P$

Proofs: All formulas are instances of the previous Lemma 128.

∎

Definition 145 *A transition system $\mathcal{T} = (S, R, v)$ is called* deterministic *if R is a function, i.e. $R(s, s_1) \wedge R(s, s_2)$ implies $s_1 = s_2$.*
As a consequence for every state s in a deterministic transition system there is a unique infinite path π^s with $\pi^s(0) = s$.
Deterministic transition systems are sometimes also called single-path *or* linear *systems.*

Lemma 130 *Let F be a CTL* formula. By F^d we denote the formula that arises from F by simply dropping all quantifiers. Thus e.g., $(\mathbf{AFAG}p)^d = \mathbf{FG}p$.*
Let $\mathcal{T} = (S, R, v)$ be a deterministic transition system. Then for all states s and all paths π:

$$\begin{array}{lll} (\mathcal{T}, s) \models F \;\Leftrightarrow\; (\mathcal{T}, \pi^s) \models F^d & \text{if } F \text{ is a state formula} \\ & \text{and } \pi^s \text{ the unique path with } \pi^s(0) = s \\ (\mathcal{T}, \pi) \models F \;\Leftrightarrow\; (\mathcal{T}, \pi) \models F^d & \text{if } F \text{ is a path formula} \end{array}$$

Proof: We proceed by induction on the complexity of F. If F is a propositional atom p, then $F^d \equiv F$ and the claim is obvious. Let us now assume the equivalence claimed by the lemma for all subformulas of F. For the cases $F \equiv F_1 \wedge F_2$, $F \equiv F_1 \vee F_2$, and $F \equiv \neg F_1$ we observe $(F_1 \wedge F_2)^d \equiv F_1^d \wedge F_2^d$, $(F_1 \vee F_2)^d \equiv F_1^d \vee F_2^d$, and $(\neg F)^d \equiv \neg F_1^d$. The inductive steps are now trivial. We consider the two case $F \equiv \mathbf{A}F_1$ and $F_1 \ \mathbf{U} \ F_2$ and leave the remaining (two) cases to the reader.

$$\begin{array}{rcll} (\mathcal{T}, s) \models \mathbf{A}F_1 & \Leftrightarrow & (\mathcal{T}, \pi^s) \models F & \text{single path property} \\ & \Leftrightarrow & (\mathcal{T}, \pi^s) \models F^d & \text{induction hypothesis} \end{array}$$

$$\begin{array}{rcl} (\mathcal{T}, \pi) \models F_1 \ \mathbf{U} \ F_2 & \Leftrightarrow & \text{there is } n \geq 0 \text{ with } (\mathcal{T}, \pi_n) \models F_2 \\ & & \text{and} \\ & & (\mathcal{T}, \pi_m) \models F_1 \text{ for all } 0 \leq m < n \\ & \Leftrightarrow & \text{there is } n \geq 0 \text{ with } (\mathcal{T}, \pi_n) \models F_2^d \\ & & \text{and} \\ & & (\mathcal{T}, \pi_m) \models F_1^d \text{ for all } 0 \leq m < n \\ & \Leftrightarrow & (\mathcal{T}, \pi) \models F_1^d \ \mathbf{U} \ F_2^d \\ & \Leftrightarrow & (\mathcal{T}, \pi) \models (F_1 \ \mathbf{U} \ F_2)^d \end{array}$$

∎

We will next take up the task of comparing the expressive power of the various temporal logics: LTL, CTL, CTL*. A principle obstacle at first are the different semantics domains: LTL formulas are evaluated in omega structures, Definition 132, while CTL and CTL* formulas are evaluated in transition structures, Definitions 141 and 144. To overcome this problem we extend the semantics of LTL formulas and define an LTL formula ϕ to be true in a state s of a transition structure $\mathcal{T} = (S, R, v)$ if ϕ is true for every path π of $\mathcal{T}$ starting in s. Since any path π may be easily considered as an omega structure, this stipulation makes sense. An LTL formula ϕ is thus said to be equivalent to a CTL* state formula ψ if $\mathbf{A}\phi \leftrightarrow \psi$ is true in all states of all transition systems. Also ϕ is said to be equivalent to a CTL* path formula ψ if $\phi \leftrightarrow \psi$ is true in all paths of all transition systems.

As observed above paths in a transition structure can easily be viewed as omega structures and we agreed to interpret LTL formulas directly for path π. Lemma 122 states an important property of LTL formulas in terms of Büchi automaton. The following lemma gives a formulation of the same fact in terms of transition structures.

Lemma 131 *Let $\mathcal{T} = (S, s_0, R, v)$ be a transition structure, π a path in $\mathcal{T}$ and ϕ an LTL formula.*
If $\pi \models \phi$ then there is a cyclic path of the form xy^ω in $\mathcal{T}$ that also satisfies $xy^\omega \models \phi$.

Proof Let PVar be the set of propositional atoms used in π. Our plan is to define a Büchi automaton $\mathcal{B} = (S, V, s_0, \delta, F)$, with $V = \mathcal{PP}(\mathrm{PVar})$ such that $L^\omega(\mathcal{B})$ conincides with the set of paths in $\mathcal{T}$. If that succeeds we can simply appeal to Lemma 122 and are finished. The set of states S of $\mathcal{B}$ is the same as for $\mathcal{T}$ and every state is also a final state $S = F$ and also the initial states coincide. For $s \in S$ let $w_s = \{p \mid v(s,p) = \mathbf{true}\} \in V$. $\delta(s, w) = \{s' \mid w = w_s \text{ and } R(s, s')\}$. The automaton $\mathcal{B}$ thus defined serves the purpose.

∎

A simple line of attack to find out if a CTL* state formula ψ is equivalent to an LTL formula would be to form ψ^d first (see Lemma 130), which obviously is an LTL formula, and then check wether ψ is equivalent to ψ^d. The next Lemma tells us that this simplistic approach is indeed the best we have; if ψ is at all equivalent to an LTL formula, then it is already equivalent to ψ^d.

Lemma 132 (CTL* vs LTL)
Let F be a CTL state formula.*
Then F is expressible in LTL iff F is equivalent to $\mathbf{A}(F^d)$.

Proof Adapted from [Clarke & Draghicescu, 1988].
Since F^d is an LTL formula the implication from right to left is obvious.
Now, we assume that F is equivalent to $\mathbf{A}(F_1)$ for an LTL formula F_1. Let $\mathcal{T} = (S, R, v)$ be an arbitrary transition system and s an arbitrary state $s \in S$. In the end we want to have $(\mathcal{T}, s) \models F \Leftrightarrow (\mathcal{T}, s) \models \mathbf{A}(F^d)$ Let us first give an overview of the whole argument and then come back to fill in the missing definitions and explanations.

1	$(\mathcal{T}, s) \models F$	iff	$(\mathcal{T}, \pi) \models F_1$ for all paths π in $\mathcal{T}$ starting in s
2		iff	$(\mathcal{T}, xy^\omega) \models F_1$ for all paths xy^ω in $\mathcal{T}$ starting in s
3		iff	$(\mathcal{T}(xy^\omega), xy^\omega) \models F_1$ for all paths xy^ω in $\mathcal{T}$ starting in s
4		iff	$(\mathcal{T}(xy^\omega), s) \models F$ for all paths xy^ω in $\mathcal{T}$ starting in s
5		iff	$(\mathcal{T}(xy^\omega), xy^\omega) \models F^d$ for all paths xy^ω in $\mathcal{T}$ starting in s
6		iff	$(\mathcal{T}, xy^\omega) \models F^d$ for all paths xy^ω in $\mathcal{T}$ starting in s
7		iff	$(\mathcal{T}, \pi) \models F^d$ for all paths π in $\mathcal{T}$ starting in s
8		iff	$(\mathcal{T}, s) \models \mathbf{A}(F^d)$

line 1 is the semantics definition of the **A** quantifier.
line 2 follows by Lemma 131
line 3 This needs some preparation. First, we need to define the transition structure $\mathcal{T}(xy^\omega) = (S_1, R_1, v_1)$. If $x = s_0 = s, \ldots, s_{i-1}$ and $y = s_i, \ldots s_k$ then $S_1 = \{0, \ldots, k\} \subset \mathbb{N}$. Further, for $j \in S$, $p \in \mathrm{PVar}$ we set $v_1(j, p) = \mathbf{true} \Leftrightarrow v(s_j, p) = \mathbf{true}$. Finally for $j_1, j_2 \in S$ let $R(j_1, j_2)$ iff $j_2 = j_1 + 1$. By construction xy^ω is the only path in $\mathcal{T}(xy^\omega)$, and that is of course the whole point of it. From the equivalence in line 2 that in line 3 follows trivially since the evaluation of the LTL formula F_1 only depends on the path xy^ω.
line 4 This follows from the previous line since xy^ω is the only path of $\mathcal{T}(xy^\omega)$.
line 5 TLemma 130 applied to the deterministic transition structure $\mathcal{T}(xy^\omega)$.
line 6 Follows from the previous equivalence since the evaluation of the LTL formula F^d only depend on the path xy^ω. This is the same argument as for the step from line 2 to 3.
line 7 Lemma 131
line 8 semantics of **A**.

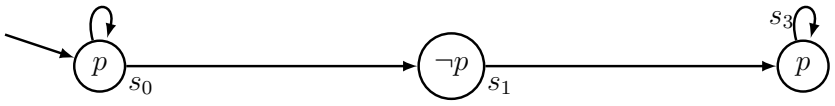


Figure 5.7: Transition system for **AFAG**p

As an application of Lemma 132 we look at the formula $\phi = \mathbf{AFAG}p$. Then $\phi^d = \mathbf{FG}p$. Thus ϕ would be expresible in LTL iff $\mathbf{AFAG}p \leftrightarrow \mathbf{AFG}p$. For the transition system in Figure 5.7 the set of all paths starting in s_0 is $\{s_0^n s_1 s_3^\omega \mid n \geq 1\} \cup \{s_0^\omega\}$. For each path, either s_1 does not occur, or eventuall s_3 is reached. Thus we have $s_0 \models \mathbf{AFG}p$. But, for the path $\pi = s_0^\omega$ we have $\pi \not\models \mathbf{FAG}p$ thus $s_0 \not\models \mathbf{AFAG}p$.

5.5 CTL Model Checking

Fixed Points

The theory of fixed points can be presented on different levels of abstraction. The most abstract version would deal with functions on a complete lattice. We will restrict our attention to establishing the terminology needed for CTL model checking. Thus it suffices to consider functions $f : \mathcal{P}(G) \to \mathcal{P}(G)$ that take as arguments a subset of a set G and return again a subset of G. Typically G will also be finite.

Definition 146 (Monotone Functions)
Let G be an arbitrary set, let $\mathcal{P}(G)$ denote the power set *of G, i.e., the set of all subsets of G.*
A function $f : \mathcal{P}(G) \to \mathcal{P}(G)$ is called monotone if for all $X, Y \subseteq G$

$$X \subseteq Y \Rightarrow f(X) \subseteq f(Y)$$

Definition 147 (Fixed Points)
Let $f : \mathcal{P}(G) \to \mathcal{P}(G)$ be a set valued function and Z a subset of G.

1. *Z is called a* fixed point *of f if $f(Z) = Z$.*
2. *Z is called the* least fixed point *of f is Z is a fixed point and for all other fixed points U of f the relation $Z \subseteq U$ is true.*
3. *Z is called the* greatest fixed point *of f is Z is a fixed point and for all other fixed points U of f the relation $U \subseteq Z$ is true.*

By $f^n(M)$ we denote as usual the n-fold iteration of f, i.e.,. $f^1(M) = f(M)$, $f^{n+1}(M) = f(f^n(M))$.

Lemma 133
Let $f : \mathcal{P}(G) \to \mathcal{P}(G)$ be a monotone function on a finite *set G.*

1. *There is a least and a greatest fixed point of f.*
2. *$\bigcup_{n \geq 1} f^n(\emptyset)$ is the least fixed point of f.*
3. *$\bigcap_{n \geq 1} f^n(G)$ is the greatest fixed point of f.*

Proofs It obviously suffices to prove 2. and 3.
(2) Monotonicity of f yields

$$\emptyset \subseteq f(\emptyset) \subseteq f^2(\emptyset) \subseteq \ldots \subseteq f^n(\emptyset) \subseteq \ldots$$

Since G is finite there must be an i such that $f^i(\emptyset) = f^{i+1}(\emptyset)$.
Then $Z = \bigcup_{n\geq 1} f^n(\emptyset) = f^i(\emptyset)$ is a fixed point of f:

$$f(Z) = f(f^i(\emptyset)) = f^{i+1}(\emptyset) = f^i(\emptyset) = Z$$

Let U be another fixed point of f. From $\emptyset \subseteq U$ we infer by monotonicity of f at first $f(\emptyset) \subseteq f(U) = U$. By induction on n we conclude $f^n(\emptyset) \subseteq U$ for all n. Thus also $Z = f^i(\emptyset) \subseteq U$.
(3) Is proved analogously.

∎

The following material in this subsection is not relevant for model checking applications. But, it is of theoretical interest, if Lemma 133 in its present form or after some adaptations is also true for infinite sets G.

Definition 148 (Continuity)
A function $f : \mathcal{P}(G) \to \mathcal{P}(G)$ is called

1. **$\cup$-continuous** *(upward continuous), if for every ascending sequence $M_1 \subseteq M_2 \subseteq \ldots \subseteq M_n \subseteq \ldots$*

$$f(\bigcup_{n\geq 1} M_n) = \bigcup_{n\geq 1} f(M_n)$$

2. **$\cap$-continuous** *(downward continuous) , if for every descending sequence $M_1 \supseteq M_2 \supseteq \ldots \supseteq M_n \supseteq \ldots$*

$$f(\bigcap_{n\geq 1} M_n) = \bigcap_{n\geq 1} f(M_n)$$

Every upward continuous or downward continuous function f is also monotonic. This can be easily seen by considering the sequence $M = M_1$ und $N = M_n$ for all $n > 1$. The reverse implication need not be true for infinite domains: there are monotonic functions that are not continuous.

For continuous functions Lemma 133 is true even in the infinite case.

Lemma 134
Let $f : \mathcal{P}(G) \to \mathcal{P}(G)$ be an upward continuous functions and $g : \mathcal{P}(G) \to \mathcal{P}(G)$ a downward continuous function.

The for all $M, N \in \mathcal{P}(G)$ such that $M \subseteq f(M)$ and $g(N) \subseteq N$ the following is true.

1. *$\bigcup_{n\geq 1} f^n(M)$ is the least fixed point of f containing M,*
2. *$\bigcap_{n\geq 1} g^n(N)$ is the greatest fixed point of g contained in N.*

Proof
ad 1: By monotonicity we first obtain

$$M \subseteq f(M) \subseteq f^2(M) \subseteq \ldots \subseteq f^n(M) \subseteq \ldots$$

Let $P = \bigcup_{n\geq 1} f^n(M)$. This immediately gives $M \subseteq P$. Furthermore

$$\begin{array}{rcll} f(P) & = & f(\bigcup_{n\geq 1} f^n(M)) & \\ & = & \bigcup_{n\geq 1} f^{n+1}(M) & \text{by continuity} \\ & = & \bigcup_{n\geq 1} f^n(M) & \text{since } f(M) \subseteq f^2(M) \\ & = & P & \end{array}$$

Assume now that Q is another fixed point of f satisfying $M \subseteq Q$. By Monotonicity and the fixed point property $f(M) \subseteq f(Q) = Q$ and furthermore for every $n \geq 1$ also $f^n(M) \subseteq Q$. Thus we obtain $P = \bigcup_{n\geq 1} f^n(M) \subseteq Q$

ad 2: analogously.

∎

Even monotone functions on infinite sets do have fixed points. This is the result of the following much quoted Knaster-Tarski-Theorem:

Theorem 135 (Knaster-Tarski-Fixed-Points Theorem)
Let $f : \mathcal{P}(G) \to \mathcal{P}(G)$ be a monotone function.
f has a least and a greatest fixed point.

Proof Let $L = \{S \subseteq G \mid f(S) \subseteq S\}$. Thus, e.g., $G \in L$. Let $U = \bigcap L$. We want to show $f(U) = U$. For all $S \in L$ we have by the definition of the intersection $\bigcap$ that $U \subseteq S$. By monotonicity and definition of L we obtain $f(U) \subseteq f(S) \subseteq S$. Thus $f(U) \subseteq U = \bigcap L$ and we have already established half of our claim. By monotonicity we get from $f(U) \subseteq U$ also $f(f(U)) \subseteq f(U)$ which yields $f(U) \in L$ and futhermore $U \subseteq f(U)$. We thus have indeed $U = f(U)$. Now assume W is another fixed point of f,i.e., $f(W) = W$. This yields $W \in L$ and $U \subseteq W$. Thus U is the least fixed point of f.
Following the same line of argument one can show that $\bigcup\{S \subseteq G \mid S \subseteq f(S)\}$ is the greatest fixed point of f.

∎

Alternative Proof of Theorem 135 using ordinals (see Section 2.5).
We reserve the symbol λ to denote limit ordinals (Def. 20 on page 33).
We define by ordinal recursion.

$$\begin{aligned} L_0 &= \emptyset \\ L_{\alpha+1} &= f(L_\alpha) \\ L_\lambda &= \bigcup_{\alpha<\lambda} L_\alpha \end{aligned}$$

We prove as a first step

$$\forall\beta(L_\beta \subseteq L_{\beta+1}) \tag{5.2}$$

If $\beta = 0$ we easily get $L_0 = \emptyset \subseteq L_1$.
If β is itself a successor ordinal $\beta = \beta_0 + 1$. The induction hypothesis is $L_{\beta_0} \subseteq L_{\beta_0+1}$ and monotonicity yield $L_\beta = f(L_{\beta_0}) \subseteq f(L_{\beta_0+1}) = L_{\beta+1}$.
It remains the case the β is a limit ordinal. From $L_\gamma \subseteq \bigcup_{\gamma<\beta} L_\gamma = L_\beta$ we obtain by monotonicity $L_{\gamma+1} = f(L_\gamma) \subseteq f(L_\beta) = L_{\beta+1}$ and thus $L_\beta = \bigcup_{\gamma<\beta} L_\gamma \subseteq \bigcup_{\gamma<\beta} L_{\gamma+1} \subseteq L_{\beta+1}$.
We are ready now to show

$$\forall\alpha\forall\beta(\alpha \leq \beta \rightarrow L_\alpha \subseteq L_\beta) \tag{5.3}$$

The proof is by transfinite induction on β. For $\beta = 0$ we only have to consider the absolutely trivial case $\alpha = \beta$.

If $\beta = \beta_0 + 1$ and $\alpha \leq \beta$ then we either have the trivial case $\alpha = \beta$ or $\alpha \leq \beta_0$. Then $L_\alpha \subseteq L_{\beta_0}$ by induction hypothesis and $L_{\beta_0} \subseteq L_{\beta_0+1} = L_\beta$ by (5.2).

If β is a limit ordinal and $\alpha < \beta$ we get directly form the definition $L_\alpha \subseteq \bigcup_{\gamma<\beta} L_\gamma = L_\beta$.

$$\text{If } \forall\alpha(\alpha < \beta \to L_\alpha \subset L_{\alpha+1}) \text{ then } L_\beta \gtrsim \beta. \tag{5.4}$$

For the definition of $a \gtrsim b$ (b is a smaller set than a) see Definition 24 on page 41. We define an injective function $g : \beta \to L_\beta$ by $g(\alpha)$ is an arbitrary element in $L_{\alpha+1} \setminus L_\alpha$. Since $\beta = \{\alpha \mid \alpha < \beta\}$ this suffices. Since $L_\beta \subseteq G$ (5.4) implies If $\forall\alpha(\alpha < \beta \to L_\alpha \subset L_{\alpha+1})$ then $G \gtrsim \beta$. But, there are ordinal β with $\beta \succ G$. Thus there must be some α with $L_\alpha = L_{\alpha+1}$. Now, L_α is a fixed point $f(L_\alpha) = L_{\alpha+1} = L_\alpha$.

∎

Example 37 *Let r be a binary relation on a set D. We may view r as a subset of the cartesian products $D^2 = D \times D$, i.e., $r \subseteq S^2$.*
The transitive, symmetric closure is usually defined as the smallest relation r_{tc} such that $r_{tc}(d, d)$ for all $d \in D$ and whenever $r_{tc}(a, b)$ and $r(b, c)$ then also $r_{tc}(a, c)$.
To relate this definition to fixed point theory we define the operator TC_r on subsets of D^2 by

$$\begin{array}{rcl} TC_r(X) & = & \{(d,d) \mid d \in D\} \quad \cup \\ & & \{(a,c) \mid \text{ there exists } b \in D \text{ with } (a,b) \in X \text{ and } r(b,c)\} \end{array}$$

It can be easily seen that r_{tc} is the least fixed point of r.

Example 38 *Consider a deterministic finite automaton $\mathcal{A} = (S, s_0, F, \Sigma, \delta)$ with finite set of states S, initial state $s_0 \in S$, set of final states $F \subseteq S$, alphabet Σ and transition function $\delta : S \times \Sigma \to S$. As usual we extend the transition function to $\delta : S \times \Sigma^* \to S$ by*

$$\begin{array}{rcl} \delta(s, \epsilon) & = & s \\ \delta(s, aw) & = & \delta(\delta(s, a), w) \\ & & \text{with } w \in \Sigma^*, a \in \Sigma \end{array}$$

The following binary relation $\equiv_{min}$ on S is of particular interest:

$$\begin{array}{rcl} s_1 \equiv_{min} s_2 & \Leftrightarrow & \text{for all } w \in \Sigma^* \\ & & \delta(s_1, w) \in F \text{ iff } \delta(s_2, w) \in F \end{array}$$

The quotient automaton $\mathcal{A}/\equiv_{min}$ is the minimal automaton equivalent to $\mathcal{A}$. But, how can we compute $\equiv_{min}$? Its definition involves the infinite set Σ^*. This where fixed points come in.

Consider the following function $F : \mathcal{P}(S \times S) \to \mathcal{P}(S \times S)$

$$\begin{aligned} H(R) \;=\; \{(s_1, s_2) \mid\; & s_1 \in F \Leftrightarrow s_2 \in F \text{ and} \\ & \text{for all } a \in \Sigma(\ (\delta(s_1, a), \delta(s_2, a)) \in R)\} \end{aligned}$$

Obviously, H is monotone, i.e. $R_1 \subseteq R_2$ implies $H(R_1) \subseteq H(R_2)$.

By Lemma 133 there is a greatest fixed point $\equiv_0$ for H, i.e., $H(\equiv_0) = \equiv_0$.

We set out to show $\equiv_{min} = \equiv_0$.
First *$\equiv_{min} \subseteq \equiv_0$ is proved by showing that $\equiv_{min}$ is a fixed point for H and using the fact that $\equiv_0$ is the greatest fixed point of H.*

$$\begin{aligned} s_1 \equiv_{min} s_2 \;&\Leftrightarrow\; \text{for all } w \in \Sigma^*(\delta(s_1, w) \in F \text{ iff } \delta(s_2, w) \in F) \\ &\quad (\text{this is the definition of } \equiv_{min}) \\ &\Leftrightarrow\; s_1 \in F \text{ iff } s_2 \in F \text{ and} \\ &\quad \text{for all } a \in \Sigma \text{ for all } w \in \Sigma^*(\delta(s_1, aw) \in F \text{ iff } \delta(s_2, aw) \in F) \\ &\quad (\text{separate the case } w = \epsilon \text{ from } w \neq \epsilon) \\ &\Leftrightarrow\; s_1 \in F \text{ iff } s_2 \in F \text{ and for all } a \in \Sigma(\delta(s_1, a) \equiv_{min} \delta(s_2, a)) \\ &\quad (\text{this makes use of } \delta(s_i, aw) = \delta(\delta(s_i, a), w)) \\ &\Leftrightarrow\; s_1 \equiv_{H(\equiv_{min})} s_2 \end{aligned}$$

Second *$\equiv_0 \subseteq \equiv_{min}$ is proved. Since $\equiv_0$ is a fixed point of H we have*

$$\begin{aligned} &\text{for all } s_1, s_2 \in S\ (s_1 \equiv_0 s_2) \\ &\text{implies} \\ &(s_1 \in F \text{ iff } s_2 \in F) \text{ and for all } a \in \Sigma(\delta(s_1, a) \equiv_0 \delta(s_2, a)) \end{aligned} \tag{5.5}$$

We will proceed by induction on n to show

$$\begin{aligned} &\text{for all } n \in \mathbb{N} \\ &\text{for all } s_1, s_2 \in S\ (s_1 \equiv_0 s_2) \\ &\text{implies} \\ &\text{for all } w \in \Sigma^* \text{ with } len(w) = n\ (\delta(s_1, w) \in F \Leftrightarrow \delta(s_2, w) \in F) \end{aligned} \tag{5.6}$$

In the initial case $n = 0$, we have to consider words w with $len(w) = 0$. Only the empty word $w = \epsilon$ satisfies this constraint and the claim follows directly from 5.5, since $\delta(s_i, \epsilon) = s_i$. For the induction step assume that 5.6 is true

for n and aim to show that it for $n+1$. So we fix $s_1, s_2 \in S$ with $s_1 \equiv_0 s_2$. Any word w' with $len(w') = n+1$ can be written as $w' = aw$ with $len(w) = n$. From 5.5 we obtain $\delta(s_1, a) \equiv_0 \delta(s_2, a)$. Using the induction hypothesis for $\delta(s_1, a), \delta(s_2, a)$ in place of s_1, s_2 we obtain

$$\delta(\delta(s_1, a), w) \in F \Leftrightarrow \delta(\delta(s_2, a), w) \in F$$

which yields

$$\delta(s_1, w') \in F \Leftrightarrow \delta(s_2, w') \in F$$

as desired. All together we have shown $\equiv_{min} = \equiv_0$ and we may use the algorithm used in the proof of Lemma 133 to compute the fixed point $\equiv_0$ to compute $\equiv_{min}$.

Example 39 *This example is a variation of Example 38 and is also taken from the theory of regular languages. Let Σ be a finite alphabet and L an arbitrary language, i.e., a set of words $L \subseteq \Sigma^*$. The following relation between words*

$$w_1 \equiv_L w_2 \quad \Leftrightarrow \quad \text{for all } u \in \Sigma^* \ (w_1 u \in L \text{ iff } w_2 u \in L)$$

plays an important role, since L is regular exactly when $\equiv_L$ has finite index.

We propose to consider the function $G_L : \mathbf{P}(\Sigma^* \times \Sigma^*) \to \mathbf{P}(\Sigma^* \times \Sigma^*)$ defined by

$$\begin{aligned} G_L(R) \quad = \quad \{(w_1, w_2) \mid \quad & (w_1 \in L \text{ iff } w_2 \in L) \text{ and} \\ & \text{for all } a \in \Sigma \ ((w_1 a, w_2 a) \in R)\} \end{aligned}$$

It can be easily seen that $G_L(R)$ is monotone, i.e., $R_1 \subseteq R_2$ implies $G_L(R_1) \subseteq G_L(R_2)$. By theorem 135 there exists a greatest fixed point E_L of G_L.

The following variation of the argument from Example shows that $\equiv_L$ equals the E_L.

$$\begin{aligned} w_1 \equiv_L w_2 \quad & \Leftrightarrow \quad \text{for all } u \in \Sigma^* \ (w_1 u \in L \text{ iff } w_2 u \in L) \\ & \quad (\text{this is the definition}) \\ & \Leftrightarrow \quad (w_1 \in L \text{ iff } w_2 \in L) \text{ and} \\ & \quad \text{for all } a \in \Sigma \text{ and all } u \in \Sigma^* \ (w_1 a u \in L \text{ iff } w_1 a u \in L) \\ & \quad (\text{distinguishing the cases } u = \epsilon \text{ and } u \neq \epsilon) \\ & \Leftrightarrow \quad (w_1 \in L \text{ iff } w_2 \in L) \text{ and for all } a \in \Sigma (w_1 a \equiv_L w_2 a) \\ & \quad \text{definition of } \equiv_L \text{ again} \\ & \Leftrightarrow \quad (w_1, w_2) \in G_L(\equiv_L) \\ & \quad (\text{definition of } G_L) \end{aligned}$$

Thus $G_L(\equiv_L) = \equiv_L$ *and therefore* $\equiv_L \subseteq E_L$.

It remains to show $E_L \subseteq \equiv_L$. *To this end we will prove by induction on* $n \in \mathbb{N}$

$$\begin{array}{l} \textit{for all } n \in \mathbb{N} \textit{ for all } w_1, w_2 \in \Sigma^*(\ (w_1, w_2) \in E_L \\ \Longrightarrow \\ \textit{for all } u \in \Sigma^* \textit{ with } len(u) = n\ (w_1 u \in L \textit{ iff } w_2 u \in L)) \end{array}$$

The case $n = 0$ *we have to derive* $w_1 \in L$ *iff* $w_1 \in L$ *from* $(w_1, w_2) \in E_L$. *By the fixed point property of* E_L *we also have* $(w_1, w_2) \in G_L(E_L)$. *Now the claim follows directly from the definition of* G_L.

So let us assume that the claim is true for n. *To prove that the claim is also true for* $n+1$ *we fix* w_1, w_2 *with* $(w_1, w_2) \in E_L$ *and* $u \in \Sigma^*$ *with* $len(u) = n+1$. *We need to arrive at* $(w_1 u \in L$ *iff* $w_2 u \in L)$. *We may write* $u = au'$ *for appropriate* $a \in \Sigma$ *and* $u' \in \Sigma^*$. *Since* $G_L(E_L) = E_L$ *we get from* $(w_1, w_2) \in E_L$ *also* $(w_1 a, w_2 a) \in E_L$. *Induction hypothesis applied for* $w_i a$ *in place of* w_i *yields* $(w_1 a)u' \in L$ *iff* $(w_2 a)u' \in L$. *Associativity of word composition yields* $w_1(au') \in L$ *iff* $w_2(au') \in L$ *as desired.*

In total we now have indeed shown $\equiv_L = E_L$.

`to be completed`

The CTL Model Checking Algorithm

The task to be solved by a model checking algorithm takes as input a transition system $\mathcal{T} = (S, R, v)$ and a CTL formula F. The output is the set $\tau(F) \subseteq S$ of states satisfying F: $\tau(F) = \{s \in S \mid s \models F\}$. For Boolean connectives this is easy. E.g., $\tau(F_1 \wedge F_2) = \tau(F_1) \cap \tau(F_2)$ and it is not so hard to find an algorithm that computes the intersection of two finite sets. More difficult are the cases where F starts with a top-level temporal operator. This problem will be solved by defining for every CTL formula F a set-valued function $f_F : \mathcal{P}(S) \to \mathcal{P}(S)$ such that

$$\begin{array}{rcl} \tau(F) = \{s \in S \mid s \models F\} & = & \text{the least fixed point of } f_F \\ & & \textit{or} \\ & & \text{the greatest fixed point of } f_F \end{array}$$

The choice whether the least or greatest fixed point is to be used depends on the top-level operator of F. We have already used in the previous text

without much ado the symbol $\tau(F)$. This notation will frequently recur in the following. So, we turn it into an official definition and give it a name.

Definition 149 (Characteristic Region)
Let $\mathcal{T} = (S, R, v)$ be a transition system and F an CTL formula. The set

$$\tau(F) = \{s \in S \mid s \models F\}$$

is called the characteristic region *of F in $\mathcal{T}$.*

Already at this early stage of explanation of the algorithm its bottleneck is apparent. The algorithm needs to work on the set of all states. It does not start with looking at one state and explore others as needed. If the state space exceeds available memory model checking is out of luck. Unfortunately, this situation arises frequently and has been dubbed the *state explosion* problem. A lot of research went into methods to alleviate the state explosion problem.

We return to the explanation of the model checking algorithm. The set-valued function f_F will, of course, be computed inductively following the structure of the formula F. If F_1, F_2 are the immediate subformulas of F then f_F will depend on $\tau(F_1)$ and $\tau(F_2)$ and in addition on the two functions f_{AX} and f_{EX} that depend on the transition relation R.

Definition 150 (Next Step Functions)
Let $\mathcal{T} = (S, R, v)$ be a given transition system.
The universal and existential next step functions $f_{AX}, f_{EX} : \mathcal{P}(S) \to \mathcal{P}(S)$ are defined by

$$\begin{aligned} f_{AX}(Z) &= \{s \in G \mid \textit{for all } t \textit{ with } sRt \textit{ we get } t \in Z\} \\ f_{EX}(Z) &= \{s \in G \mid \textit{there exists a } t \textit{ with } sRt \textit{ and } t \in Z\} \end{aligned}$$

Thus $f_{AX}(Z)$ is the set of all states with all next states in Z, while $f_{EX}(Z)$ is the set of all states with one next state in Z.

We are now ready to present the model checking algorithm.

Definition 151 (CTL Model Checking Algorithm)
Let $\mathcal{T} = (S, R, v)$ be a transition system and F an CTL formula. The characteristic region $\tau(F)$ is computed by the following high-level recursive

algorithm:

1	$\tau(p)$	$=$	$\{s \in S \mid v(s,p) = \mathbf{true}\}$
2	$\tau(F_1 \wedge F_2)$	$=$	$\tau(F_1) \cap \tau(F_2)$
3	$\tau(F_1 \vee F_2)$	$=$	$\tau(F_1) \cup \tau(F_2)$
4	$\tau(\neg F_1)$	$=$	$S \setminus \tau(F_1)$
5	$\tau(\mathbf{A}(F_1\ \mathbf{U}\ F_2))$	$=$	$\mathrm{lfp}[\tau(F_2) \cup (\tau(F_1) \cap f_{AX}(Z))]$
6	$\tau(\mathbf{E}(F_1\ \mathbf{U}\ F_2))$	$=$	$\mathrm{lfp}[\tau(F_2) \cup (\tau(F_1) \cap f_{EX}(Z))]$
7	$\tau(\mathbf{AF}F_1)$	$=$	$\mathrm{lfp}[\tau(F_1) \cup f_{AX}(Z)]$
8	$\tau(\mathbf{EF}F_1)$	$=$	$\mathrm{lfp}[\tau(F_1) \cup f_{EX}(Z)]$
9	$\tau(\mathbf{EG}F_1)$	$=$	$\mathrm{gfp}[\tau(F_1) \cap f_{EX}(Z)]$
10	$\tau(\mathbf{AG}F_1)$	$=$	$\mathrm{gfp}[\tau(F_1) \cap f_{AX}(Z)]$

Theorem 136 (Correctness of CTL Model Checking)
The algorithm from Definition 151 is correct.

Proof Let us start from the bottom of the definition.

Case 10 AG $\mathbf{F_1}$ From the definition of the characteristic region of **AG** F_1 we easily obtain:

$$\tau(\mathbf{AG}\ F_1) = \{s \in S \mid s \in \tau(F_1) \text{ and } h \in \tau(\mathbf{AG}\ F_1) \text{ for all } h \text{ with } sRh\}$$

Using the notion from Definition 151 this can be written as

$$\tau(\mathbf{AG}\ F_1) = \tau(F_1) \cap f_{AX}(\tau(\mathbf{AG}\ F_1))$$

So, $\tau(\mathbf{AG}\ F_1)$ is a fixed point of the function $\tau(F_1) \cap f_{AX}(Z)$. It remains to see that it is the greatest fixed point. Let H be another fixed point, i.e., a subset of S satisfying $H = \tau(F_1) \cap f_{AX}(H)$. We want to show $H \subseteq \tau(\mathbf{AG}\ F_1))$.

To this end we consider some $g_0 \in H$ with the aim of showing that for all $n \geq 0$ and all g_i satisfying $g_{i-1}Rg_i$ for all $1 \leq i \leq n$ we obtain $g_n \in \tau(F_1)$. By definition that is to say $g_0 \in \tau(\mathbf{AG}F_1)$. We first observe

$\mathbf{H \subseteq \tau(F_1)}$ which follows readily from $H = \tau(F_1) \cap f_{AX}(H) \subseteq \tau(F_1)$.

Thus it suffices to show for all $n \geq 0$ that $g_n \in H$. For $n = 0$ that is true by assumptions. For the induction step we assume $g_{n-1} \in H$. From

the fixed point property of H we infer $g_{n-1} \in H \subseteq f_{AX}(H) = \{g \mid$ for all h with gRh we have $h \in H\}$. Thus $g_n \in H$.

Case 6 $\mathbf{E}(\mathbf{F_1}\ \mathbf{U}\ \mathbf{F_2})$ From the definition of the characteristic region of **AG** F_1 we easily obtain:

$$\begin{aligned}\tau(\mathbf{E}(F_1\ \mathbf{U}\ F_2)) \quad = \quad & \{s \in S \mid s \models F_2 \text{ or } s \models F_1 \text{ and} \\ & \text{there exists } h \text{ with } sRh \text{ and } h \in \tau(\mathbf{E}(F_1\ \mathbf{U}\ F_2))\}\end{aligned}$$

Using the notion from Definition 151 this can be written as

$$\tau(\mathbf{E}(F_1\ \mathbf{U}\ F_2)) = \tau(F_2) \cup (\tau(F_1) \cap f_{EX}(\tau(\mathbf{E}(F_1\ \mathbf{U}\ F_2))))$$

That is to say $\tau(\mathbf{E}(F_1\ \mathbf{U}\ F_2))$ is a fixed point of the function $\tau(F_2) \cup (\tau(F_1) \cap f_{EX}(Z))$ where Z is the parameter of the functions.

It remains to show that it is the least one. Consider thus another subset $H \subseteq S$ with $H = \tau(F_2) \cup (\tau(F_1) \cap f_{EX}(H))$. We need to convince ourselves that $\tau(\mathbf{E}(F_1\ \mathbf{U}\ F_2)) \subseteq H$ is true. To this end let $g_0 \in \tau(\mathbf{E}(F_1\ \mathbf{U}\ F_2)$ and try to arrive at $g_0 \in H$. By the semantics of $\mathbf{E}(F_1\ \mathbf{U}\ F_2)$ there is an $n \in \mathbb{N}$ and there are g_i for $1 \leq i \leq n$ satisfying

1. $g_i R g_{i+1}$ for all $0 \leq i < n$.
2. $g_n \in \tau(F_2)$.
3. $g_i \in \tau(F_1)$ for all $0 \leq i < n$.

We set out to prove $g_n \in H$ by induction on n.

$\mathbf{n = 0}$ By the fixed point property of H

$$H = \tau(F_2) \cup (\tau(F_1) \cap f_{EX}(H)) \supseteq \tau(F_2)$$

and we are through since $g_0 = g_n \in \tau(F_2)$.

$\mathbf{n - 1 \rightsquigarrow n}$ By induction hypothesis we have $g_1 \in H$ (since we get from g_1 to g_n in $n - 1$ steps). Since $g_0 \in \tau(F_1)$ and $g_0 R g_1$ we obtain $g_0 \in (\tau(F_1) \cap f_{EX}(H)) \subseteq H$ and thus also $g_0 \in H$.

There are no suprises in the proofs of the remaining cases, some of which may be found in in [Clarke *et al.*, 1993, Clarke *et al.*, 2001] .

∎

The CTL Model Checking: An Example

We want to illustrate the working of the model checking algorithm from Definition 151 by a small, yet non-trivial example. We will use the transition system $\mathcal{T}_{me2}$ from Figure 5.6, which we repeat for the readers convenience in Figure 5.8 The example will involve the propositional atoms N_1, N_2, T_1, T_2,

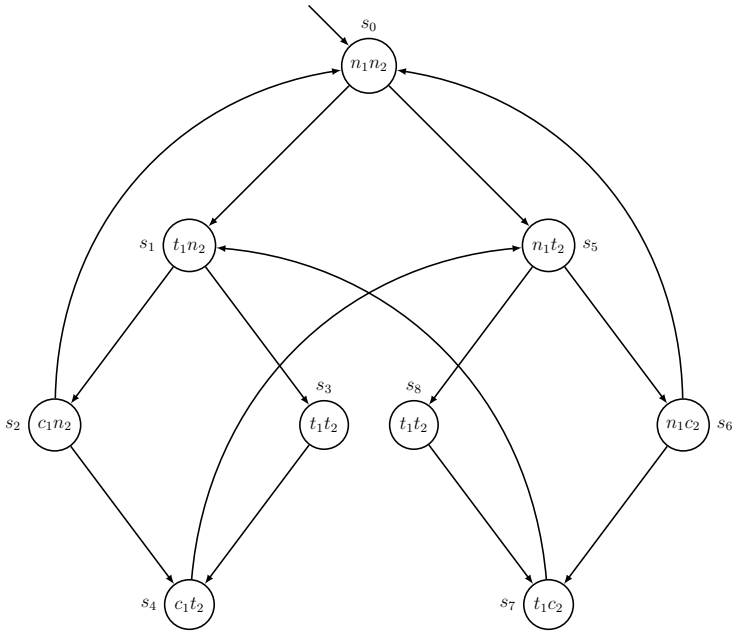


Figure 5.8: Mutual Exclusion (repeated from Figure 5.6)

C_1, and C_2. The interpretation of these atoms in the transition system is given by the following table.

	0	1	2	3	4	5	6	7	8
N_1	1	0	0	0	0	1	1	0	0
N_2	1	1	1	0	0	0	0	0	0
T_1	0	1	0	1	0	0	0	1	1
T_2	0	0	0	1	1	1	0	0	1
C_1	0	0	1	0	1	0	0	0	0
C_2	0	0	0	0	0	0	1	1	0

We want to check whether the formula

$$F = T_1 \rightarrow \mathbf{AF}C_1 \equiv \neg T_1 \vee \mathbf{AF}C_1$$

is true in state 1. Here and in the following we will write state i instead of state s_i. The recursive algorithm will successively compute the characterisitc region of the following formulas:

$$F, \neg T_1, T_1, \mathbf{AF}C_1, \text{ and } C_1$$

We will present the computations starting with the innermost subformulas first, i.e., in the order $\tau(T_1)$, $\tau(C_1)$, $\tau(\neg T_1)$, $\tau(\mathbf{AF}C_1)$, and $\tau(F)$. This will make it much easier to follow the algorithm, since when a recursice call is started, we know already its result. In the end we check $1 \in \tau(F)$.

The characteristic regions of T_1 and C_1 can be read of from the transition system $\mathcal{T}_{me2}$

$$\tau(T_1) = \{1, 3, 7\} \tag{5.7}$$

$$\tau(C_1) = \{2, 4\} \tag{5.8}$$

From which we get easily

$$\tau(\neg T_1) = \{0, 2, 4, 5, 6\} \tag{5.9}$$

The next step is to compute $\tau(\mathbf{AF}C_1)$ which according to Definition 151 amounts to the computation of the least fixed point of the set function $f(Z) = \tau(C_1) \cup f_{AX}(Z) = f_{AX}(Z) \cup \{2, 4\}$. Following Lemma 133 we compute successively $f(\emptyset)$, $f^2(\emptyset)$, $\ldots f^n(\emptyset)$ till we reach a stationary value, i.e. $f^n(\emptyset) = f^{n+1}(\emptyset)$.

$$\begin{array}{lcl} f^1(\emptyset) & = & \{2, 4\} \\ f^2(\emptyset) & = & \{2, 3, 4\} \\ f^3(\emptyset) & = & \{1, 2, 3, 4\} \\ f^4(\emptyset) & = & \{1, 2, 3, 4, 7\} \\ f^5(\emptyset) & = & \{1, 2, 3, 4, 7, 8\} \\ f^6(\emptyset) & = & \{1, 2, 3, 4, 7, 8\} \end{array}$$

Thus

$$\tau(\mathbf{AF}C_1) = \{1, 2, 3, 4, 7, 8\} \tag{5.10}$$

and finally

$$\tau(F) = \tau(\neg T_1) \cup \tau(\mathbf{AF}C_1) = \{0, 1, 2, 3, 4, 5, 6, 7, 8\} = S \tag{5.11}$$

Since $1 \in \tau(F)$ we conclude

$$s_1 \models F \tag{5.12}$$

5.6 Exercises

Exercise 5.6.1
Let Σ_V be the vocabulary defined in Definition 125.
In Lemma 113 it was shown that the order relation $<$ on $\mathbb{N}$ can be defined in the theory FOW of second order monadic logic. The question arises what else can be defined in this theory. Is it possible to define addition? More precisely, is there a formula $\phi(x, y, z)$ of second order monadic logic in the vocabulary Σ_V with free first-order variables x, y, z, such that for any model $\mathcal{W}$ of FOW and elements n, m, k in the universe of $\mathcal{W}$

$$\mathcal{W} \models \phi[n, m, k] \Leftrightarrow n + m = k.$$

Remember that the universe of $\mathcal{W}$ consists exactly of the elements $0^{\mathcal{W}}, (s(0))^{\mathcal{W}}, \ldots (s^n(0))^{\mathcal{W}}, \ldots$ and that we agreed to identify $(s^n(0))^{\mathcal{W}}$ with n.

Exercise 5.6.2
Show that the following LTL formulas are tautologies

1. $(F_1 \wedge F_2)\ \mathbf{U}\ G \leftrightarrow (F_1\ \mathbf{U}\ G) \wedge (F_2\ \mathbf{U}\ G)$
2. $F\ \mathbf{U}\ (G_1 \vee G_2) \leftrightarrow (F\ \mathbf{U}\ G_1) \vee (F\ \mathbf{U}\ G_2)$

Exercise 5.6.3
Formalize the following properties in LTL

1. *Every p is followed by a q, or more precisely:*
   for every time point t at which p is true there is a time point s, $s \geq t$ such that q is true at s.
2. *Every p is followe by a q but at least 3 time points later, or more precisely:*
   or every time point t at which p is true there is a time point s, $s - t \geq 3$ such that q is true at s.

Exercise 5.6.4
Consider the following property:

p occurs infinitely often. Every p is followed by a q. At the first occurence of p also q is immediately true. For every following occurence of p the time till q occurs is at least on time step longer than on the previous occurence of p. Try to formalize this property in LTL or show that this is not possible.

Exercise 5.6.5
Formalize the five properties from Example 36 on page 243 by CTL formulas assuming that there are n participants in the protocol instead of two.

Exercise 5.6.6
Show that the subset $K_{p,q} \subseteq V_{p,q}^{\omega}$ defined by

$$K_{p,q} = \{w \in V_{p,q}^{\omega} \mid \text{for all } 1 \leq i \leq p \text{ there is exactly one } k \text{ such that the } i\text{-th position in the letter } w(k) \text{ equals } 1\}$$

is an omega-regular language.

Exercise 5.6.7
Prove Lemma 117, that every homomorphic image of an omega-regular set is again omega-regular.

Exercise 5.6.8 *Compute the formulas L_1, L_2 and M_3 for the example automaton is in Lemma 124 for the formula $F \equiv \mathbf{GF}\neg p$.*

Exercise 5.6.9 *Prove the following generalization of Theorem 135:*
Let $f : \mathcal{P}(G) \to \mathcal{P}(G)$ be a monotone function on an arbitrary, possibly infinite, set G. Let $U_0 \subseteq G$ such that $U_0 \subseteq f(U_0)$.
Then there exists a fixed point U of f with

1. *$U_0 \subseteq U$ and*
2. *For every fixed point W with $U_0 \subseteq W$ we have $U \subseteq W$*

i.e., W is the least fixed point above U_0.

Exercise 5.6.10 *This is a follow-up on Exercise 5.6.9.*
What would be the requirement for a set U_0 such that there exists a greatest fixed point below U_0?

Exercise 5.6.11 *Show that the formula* **AGEF**p *is not expressible in LTL. Hint: Obviously Lemma 132 has to be used.*

Exercise 5.6.12 *Let F be a monotone operator on subset of the set D. Define $G(X) = D \setminus F(D \setminus X)$. Show*

1. *G is monotone.*
2. *The least fixed point $lfp(G)$ of G is $D \setminus gfp(F)$.*
3. *The greatest fixed point $gfp(G)$ of G is $D \setminus lfp(F)$.*

Exercise 5.6.13 *Let $\mathcal{B} = (S, V_{p,q}, s_0, \delta, F)$ be a Büchi automaton with edge vocabulary $V_{p,q}$, see Definition 126 on page 217 such that*

$$L^{\omega}(\mathcal{B}) = \{w \in K_{p,q} \mid w \models \phi[x_1, \ldots, x_p, X_1, \ldots, X_q]\}$$

for some monadic second-order formula ϕ.

Construct a Büchi automaton $\mathcal{B}_{ex}$ such that

$$L^{\omega}(\mathcal{B}_{ex}) = \{w \in K_{p,q} \mid w \models \exists x_p \phi[x_1, \ldots, x_{p-1}, X_1, \ldots, X_q]\}$$

This is a constructive version of part of the argument in the proof of Theorem 114.

Chapter 6

Solutions to Exercises

Solutions to Chapter 1

Exercise 1.2.1 Let $\{c_i \mid i \in \mathbb{N}\}$ be new constant symbols. By assumption the set

$$\Gamma = \{c_i \neq c_j \mid i, j \in \mathbb{N} \text{ with } i \neq j\} \cup \{\neg F\}$$

is inconsistent. By the compactness theorem of first-order logic there is a finite subset $\Gamma_0 \subset \Gamma$ that is also inconsistent. Let k be such that $\Gamma_0 \subseteq \{c_i \neq c_j \mid i, j \leq k \text{ with } i \neq j\} \cup \{\neg F\}$. As a superset of an inconsistent set this set is also inconsistent. Furthermore, since the part $\{c_i \neq c_j \mid i, j \leq k \text{ with } i \neq j\}$ is satisfiable in any structure $\mathcal{M}$ with k or more elements, we must have $\mathcal{M} \models F$.

Exercise 1.2.2 Assume F is Σ_1-tautology. Let $\mathcal{M}$ be an arbitrary Σ_1-structure. Let $\mathcal{M}_1$ be as in Definition 2. By construction we know $\mathcal{M}_1 \models F \Leftrightarrow \mathcal{M} \models F_2$. Since F was assumed to be a tautology we have $\mathcal{M}_1 \models F$ and thus $\mathcal{M} \models F_2$, as desired
Now assume conversely that F_2 is a Σ_2-tautology. Let $\mathcal{N}$ be an arbitrary Σ_1-structure. We aim for $\mathcal{N} \models F$. By Definition 2 there is a Σ_2-structure $\mathcal{M}$ with $\mathcal{N} \simeq \mathcal{M}_1$. By assumption on F_2 we have $\mathcal{M} \models F_2$ and thus by construction $\mathcal{M}_1 \models F$. By isomorphy $\mathcal{N} \models F$.

Exercise 1.2.3 If validity of Σ_2-formulas were decidable. We could get a procedure for deciding Σ_1-formulas as follows: For a Σ_1-formula F construct F_2. If F_2 is a valid Σ_2-formula then F is a valid Σ_1-formula by Exercise 1.2.2.

Exercise 1.2.4 Let $\Sigma_2 = \Sigma_2^r \cup \Sigma_2^f$ be a given signature containing only binary relation symbols $R \in \Sigma_2^r$ and constant symbols $c \in \Sigma_2^f$.

Let $\Sigma_3 = \{rel(\_, \_, \_)\} \cup \Sigma_2^f \cup \{c_R \mid R \in \Sigma_2^r\}$.

For any Σ_2 formula F let the Σ_3 formula F^3 be obtained by replacing every atomic subformula $R(t_1, t_2)$ in F, where t_i are either constants from Σ_2^f or variables by $rel(c_R, t_1, t_2)$.

For a Σ_3-structure $\mathcal{M}$ we obtain the Σ_2-structure $\mathcal{M}^2$ by

1. the universe M of $\mathcal{M}^2$ is the same as the universe of $\mathcal{M}$

2. the interpretation of the constants $c \in \Sigma_2^f$ remains unchanged

3. for all $e_1, e_2 \in M \quad (e_1, e_2) \in R^{\mathcal{M}^2} \Leftrightarrow (c_R^{\mathcal{M}}, e_1, e_2) \in rel^{\mathcal{M}}$

An easy induction on the complexity of F shows $\mathcal{M}^2 \models F \Leftrightarrow \mathcal{M} \models F^3$.

For the second part of the exercise conisder a Σ_2-structure $\mathcal{N}$. We assume in addition that $card(N) \geq card(\Sigma_2^r)$, i.e. there are at least as many element in N as there are (binary) relation symbols in the signature Σ_2. We define the Σ_3-structure $\mathcal{M}$ by

1. the universe N of $\mathcal{M}$ is the universe of $\mathcal{N}$.

2. the interpretation of the constants in Σ_2^f remains unchanged.

3. the constants c_R for $R \in \Sigma_2^r$ are interpreted arbitrarily subject only to the condition that different constants are interpreted by different elements. This is why we need the cardinality requirement on N.

4. $(e_1, e_2, e_3) \in rel^{\mathcal{M}} \quad \Leftrightarrow \quad$ there is $R \in \Sigma_2^r$ with $c_R^{\mathcal{M}} = e_1$ and $(e_2, e_3) \in R^{\mathcal{N}}$

If $\mathcal{M}^2$ is the Σ_2-structure obtained from $\mathcal{M}$ by the construction described above it can readily be seen that $\mathcal{M}^2 = \mathcal{N}$.

It remains to discuss wether the restriction on the size of $\mathcal{N}$ affects the transfer of undecidability. Assume that the tautology property of Σ_3-formulas could be decided. Then we could also decide which Σ_3-formulas are true in all structures with more than k elements. (Let F_k be a formula that is exactly true in structures with more than k elements. F_k uses only the equality relation. Then F is true in all structures with more than k elements iff $F_k \to F$ is true in all structures.) The above argument shows that we can then decide which Σ_2-formulas are true in all structures with more than k elements. But then we can also decide which Σ_2-formulas are true in all structures by checking in addition all finitely many structures with less than k elements.

Exercise 1.2.5 Let Σ_1 be an arbitrary (unrestricted) relational vocabulary. The signature Σ_2 contains

1. All constant symbols from Σ_1

2. For every relation symbol $R \in \Sigma_1$ of arity $\neq 2$ a new constant symbols c_R
3. A binary relation symbol rel
4. Binary relation symbols rel_i^n for every i, $1 \leq i \leq n$ and $1 \leq n \leq k$, where k is the maximal arity of a relation symbol in Σ_1 or $k = \infty$ if there is no bound on the arity of relation symbol in Σ_1.

For a Σ_1-formula F_1 its translation into a Σ_2-formula F_2 is obtained by replacing every atomic subformula $R(t_1, \ldots, t_n)$ with $n \neq 2$ by

$$\exists z(rel(c_R, z) \wedge \bigwedge_{i=1}^{n} rel_i^n(z, t_i))$$

Note, that t_i is either a constant symbol or a variable and of course z is chosen to be different from all t_i.

Consider an infinite Σ_2-structure $\mathcal{M}$. The Σ_1-structure $\mathcal{M}_1$ is obtained as follows

1. The universe M of $\mathcal{M}_1$ is the same as that of $\mathcal{M}$
2. For $n \neq 2$ and R an n-ary relation symbol,$n \neq 2$, in Σ_1

$$R^{\mathcal{M}_1} = \{(a_1, \ldots, a_n) \in M^n \mid rel^{\mathcal{M}}(c_R^{\mathcal{M}}, a) \wedge \bigwedge_{i=1}^{n} rel_i^{\mathcal{M}}(a, a_i) \text{ for some } a\}$$

It is fairly obvious that for any variable assignment β

$$(\mathcal{M}_1, \beta) \models R(t_1, \ldots, t_n) \Leftrightarrow (\mathcal{M}, \beta) \models \exists z(rel(c_R, z) \wedge \bigwedge_{i=1}^{n} rel_i(z, t_i))$$

and thus

$$\mathcal{M}_1 \models F_1 \Leftrightarrow \mathcal{M} \models F_2$$

It remains to show that also the second part of Definition 2 is true. To this end consider a Σ_1-structure $\mathcal{N}$. To construct the required Σ_2-structure $\mathcal{M}$ we need some auxiliary functions. Since N, the universe of $\mathcal{N}$, is infinite there are for every n injective functions $f_n : N^n \to N$. Now $\mathcal{M}$ is given by

1. The universe of $\mathcal{M}$ is N

2. The interpretations $c_R^{\mathcal{M}}$ are chosen arbitrarily subject only to the condition that for different relation symbols R and S the interpretations are different, i.e., $c_R^{\mathcal{M}} \neq c_S^{\mathcal{M}}$.

3. $(a, b) \in rel^{\mathcal{M}}$ if there is $R \in \Sigma_1$ such that $c_R^{\mathcal{M}} = a$ and there is an n-tuple $(b_1, \ldots, b_n) \in N^n$, $n =$ the arity of R, with $f_n(b_1, \ldots, b_n) = b$ and $(b_1, \ldots, b_n) \in R^{\mathcal{N}}$.

4. $(b, c) \in (rel_i^n)^{\mathcal{M}}$ if there is an n-tuple $(b_1, \ldots, b_n) \in N^n$ with $f_n(b_1, \ldots, b_n) = b$ and $b_i = c$.

It is not hard to see that

$$\{(b_1, \ldots, b_n) \in M^n \mid rel^{\mathcal{M}}(c_R^{\mathcal{M}}, a) \wedge \bigwedge_{i=1}^{n} rel_i^{\mathcal{M}}(b, b_i) \text{ for some } b\}$$

coincides with $R^{\mathcal{N}}$ and thus $\mathcal{N} \simeq \mathcal{M}_1$.

Solutions to Chapter 2

Exercise 2.11.1 This is a simple reformulation of the Foundation Axiom making use of the appreviations $\emptyset$ and $\cap$.

Exercise 2.11.2 **(1)** The initial case $m = 0$ is trivial, since $n \in 0$ is never true. Assume $n \in m \rightarrow n^+ \in m \vee n^+ = m$
Show $n \in m^+ \rightarrow n^+ \in m^+ \vee n^+ = m^+$
There are two cases for $n \in m^+$ ot be true.
case 1 $n = m$.
This yields immediately $n^+ = m^+$ and we are finished.
case 2 $n \in m$.
By induction hypothesis we have $n^+ \in m$ or $n^+ = m$. Both cases immediately entail $n^+ \in m^+ = m \cup \{m\}$.
(2) The base case, $n = 0$, follows from Lemma 7.

Assume $(n \in m \vee n = mm \in n)$
Show $n^+ \in m \vee n^+ = m \vee m \in n^+)$
case 1 $n \in m$.

By part 1 of this exercise we get $n^+ \in m$ or $n^+ = m$. As needed. *case 2* $n = m \vee m \in n$)
Immediately implies $m \in n^+ = n \cup \{n\}$.

Exercise 2.11.3
Let $a = \{s_1, \ldots, s_k\}$ with $k > 0$. By assumption $\bigcup a = a$. We fix an arbitrary element $s_{i_0} \in a$. By definition of union there must be an index i_1 such that $s_{i_0} \in s_{i_1}$. Again there must be an index i_2 such that $s_{i_1} \in s_{i_2}$. Iterating this argument we obtain in total $k+1$ indices $i_0, i_1, \ldots i_k$ such that $s_{i_0} \in s_{i_1} \in s_{i_2} \in \ldots \in s_{i_k}$. Since there are only k elements there must be $0 \leq u < v \leq k$ with $s_{i_u} = s_{i_v}$. But now, $s_{i_u} \in s_{i_{+11}} \in \ldots \in s_{i_v} = s_{i_u}$ contradicts the axiom of foundation.

▪

Exercise 2.11.4
(1) It can be easily checked that the set $\{0, 1, 2, \{1\}, \{2\}\}$ is transitive, but its elements $\{1\}$ and $\{2\}$ are neither equal nor is one an element of the other.
(2) Assume that n is a transitive set satisfying $\forall x \forall y (x \in a \wedge y \in a \rightarrow (x \in y \vee y \in x \vee x = y)$.
The same proof as for Lemma 8(1) shows that n^+ is also transitive. It remains to show the additional property for n^+. For this we fix elements $x, y \in n^+ = n \cup \{n\}$. If $x, y \in n$ then we get from the induction hypothesis $(x \in y \vee y \in x \vee x = y)$, as desired. If $x = y = n$ we are immediately finished. So, it remains to investigate the case $x \in n \wedge y = n$ (and the symmetric one $y \in n \wedge x = n$). The first case immediatly yields $x \in y$ and the second $y \in x$.

▪

Exercise 2.11.5

1 This is the classical Russell paradox.
Assume that there is a set c with $c = \{x \mid x \notin x\}$. Then the formula $c \in \{x \mid x \notin x\}$ is be the rules for eliminating class terms logically equivalent to $c \notin c$. Thus $c \in c \leftrightarrow c \notin c$, an outright contradiction.

2 Since $\{x \mid x = x\}$ is the class term denoting all sets, this exercise asks to show that the collection of all sets is not a set.
Assume, that it is, i.e., that there is a set c with $c = \{x \mid x = x\}$. Since $c = c$ is true, we obtain $c \in c$ which contradicts the foundation axiom.

3 Assume for the sake of a contradiction that there is a set r satisfying $r = \{x \mid rel(x)\}$. The singleton set $\{\langle r, r\rangle\}$ satisfies $rel(\{\langle r, r\rangle\})$, thus $\{\langle r, r\rangle\} \in r$. Since $\langle r, r\rangle = \{\{r\}\}$ this leads to $r \in \{r\} \in \{\{r\}\} \in \{\{\{r\}\}\} \in r$ which again contradicts the foundation axiom.

▪

Exercise 2.11.6 Observe that

$$\begin{array}{lcl} z_1, z_2 & \in & a \cup b \\ \{z_1\}, \{z_1, z_2\} & \in & \mathcal{P}(a \cup b) \\ \{\{z_1\}, \{z_1, z_2\}\} & \in & \mathcal{P}(\mathcal{P}(a \cup b)) \\ \langle z_1, z_2\rangle & \in & \mathcal{P}(\mathcal{P}(a \cup b)) \end{array}$$

Thus

$$a \times b = \{x \in \mathcal{P}^2 \mid \exists z_1 \exists z_2 (x = \langle z_1, z_2\rangle \wedge z_1 \in a \wedge z_2 \in b)\}$$

Now it is easy to see that the previously proved existence of the union of two sets, the powerset and subset axioms allow to deduce the existence of the Cartesian product.

▪

Exercise 2.11.7 Assume that Ord is a set, then by Lemma 24(4) $\alpha = \bigcup Ord$ is also an ordinal, i.e., $\alpha \in Ord$. Thus also $\alpha + 1 \in Ord$. From $\alpha \in (\alpha + 1) \in Ord$ we obtain the contradiction to the foundation axiom $\alpha \in \bigcup Ord = \alpha$.

▪

Exercise 2.11.8 Let $\alpha = \bigcup x$.
For $\beta \in x$ we get by definition of $\bigcup$ that $\beta \subseteq \alpha$. Now consider an ordinal γ with $\beta \leq \gamma$ for all $\beta \in x$, i.e. $\beta \subseteq \gamma$ for all $\beta \in x$. The definition of $\bigcup$ yields immediately $\alpha = \bigcup_{\beta \in x} \beta \subseteq \gamma$.

▪

Exercise 2.11.9 Assume that there is a set a such that $a \times a = \mathcal{P}(a)$. By the foundation axiom A2 there is $b \in \mathcal{P}(a)$ such that $b \cap \mathcal{P}(a) = \emptyset$. Since $b \in a \times a$ there are elements $x, y \in a$ with $b =< x, y >= \{\{x\}, \{x, y\}\}$. By choice of b we have $\{x\} \notin \mathcal{P}(a)$. Thus $x \notin a$ contrary to the choice of x.

▪

Exercise 2.11.10 The answer is no! Remembering the definition $\langle a, b\rangle = \{\{a\}, \{a, b\}\}$ we obtain

$$s \in \{q, s\} \in \langle q, s\rangle \in r \in \{p, r\} \in \langle p, r\rangle \in s$$

This contradicts the foundation axioms.

▪

Exercise 2.11.11 An immediate application of the axiom of foundation does not yield the desired contradiction. We need an additional application of the subset axiom. This axiom yields the existence of a set a_0 with $a_0 = \{z \in a \mid z = a\} = \{a\}$. By the foundation axiom there should be an element $b \in a_0$ with $a_0 \cap b = \emptyset$. Since a is the only element of a_0, this is $a_0 \cap a = \emptyset$. But, this contradicts $a \in a_0 \cap a$.

▪

Exercise 2.11.12 Assume there are sets $a_1, \ldots a_n$ with $a_1 \in a_2 \in \ldots \in a_{n-1} \in a_n \in a_1$. Let $c = \{a_1, \ldots a_n\}$. By the foundation axiom A2 there is an element $b \in c$ with

$$\forall z(z \in c \rightarrow z \notin b)$$

Now, $b = a_{i+1}$ for $1 \leq i \leq n-1$ is not possible since we have $a_i \in c$ and $a_i \in a_{i+1}$. But, also $b = a_1$ is not possible since $a_n \in c$ and $a_n \in a_1$.

▪

Exercise 2.11.13 Assume that $H = Ord \setminus X \neq \emptyset$.

We will first show that $\bigcap H$ is again an ordinal. By Lemma 2(3) we know that $\bigcap H$ is a set. The rest of the argument is an easy extension of the proof of Lemma 24(2). To prove transitivity of $\bigcap H$ consider $x \in H$. Since

all elements of H are in particular transitive set we get for all $\alpha \in H$ that $\alpha \subseteq \alpha$. Thus also $\alpha \subseteq \bigcap H$. To check the second defining property of ordinals consider $\alpha, \beta \in \bigcap H$. By our assumptions H is not empty, so there is $\gamma \in H$ and of course also $\alpha, \beta \in \gamma$. By the ordinal property of γ we obtain $\alpha \in \beta$ or $\beta \in \alpha$ or $\beta = \alpha$. So we have established that $\bigcap H = \delta$ is an ordinal.

For all $\alpha \in \delta = \bigcap(Ord \setminus X)$ we get $\forall\beta(\beta \notin X \to \alpha \in \beta)$. This implies $\alpha \in X$, since $\alpha \notin X$ would lead to the contradiction $\alpha \in \alpha$. By the inductive property of X we get $\delta \in X$. From $\delta = \bigcap(Ord \setminus X)$ we also infer for all $\beta \notin X$ the relation $\delta \subseteq \beta$. Since we know already $\delta \in X$ we must even have $\forall\beta(\beta \notin X \to \delta \subset \beta)$. By Lemma 24(1) this implies $\forall\beta(\beta \notin X \to \delta \in \beta)$. This leads to the contradiction $\delta \in \bigcap_{beta \notin X} \beta = \delta$. Thus we must indeed have $Ord = X$.

∎

Exercise 2.11.14 The answer is no since

$$\ldots O^{k+1} < 0^k 1 < \ldots 001 < 01$$

∎

Exercise 2.11.15 We apply Lemma 22 with

$$\begin{array}{lcl} WO(x,y) & \equiv & ord(x) \wedge ord(y) \wedge x \in y \\ \phi(u) & \equiv & \forall v((u,\epsilon) \cong (v,\epsilon) \to u = v) \end{array}$$

Requirement (1) of Lemma 22 follows from the foundation axioms. To satisfy requirement (2) we start with an arbitrary set x and need to find a superset $x \subseteq u$ closed under the predecessors of WO. We claim that $u = x \cup \bigcup\{w \in x \mid w \text{ is an ordinal}\}$ does the job. Assume $z \in x$ and $WO(v,z)$ then $v \in z$ and z is an ordinal and thus $v \in z \subseteq u$. If $z \in w$ for an ordinal $w \in x$ and $WO(v,z)$ then $v \in z$ and v and z are ordinals. Since the ordinal w is in particular a transitive set we get $v \in z \subseteq w$and there for $z \in w \subseteq u$. Thus we know that Lemma 22 is applicable with the given instantiations of WO. It remains to proof the argument for the induction step for α an arbitrary ordinal:

$$\begin{array}{ll} \text{If} & \forall\beta(\beta \in \alpha \to \forall\gamma((\beta,\epsilon) \cong (\gamma,\epsilon) \to \beta = \gamma)) \\ \text{then} & \forall\gamma((\alpha,\epsilon) \cong (\gamma,\epsilon) \to \alpha = \gamma)) \end{array}$$

So we consider an isomorphism $f : (\alpha, \epsilon) \to (\gamma, \epsilon)$. For every $\beta \in \alpha$ the restriction f' of f to $\beta \subseteq \alpha$ is an isomorphism $f' : (\beta, \epsilon) \to (f(\beta), \epsilon)$. By assumption we get $\beta = f(\beta)$. This immediately yields $\alpha = \gamma$.

∎

Exercise 2.11.16 Assume, for the sake of a contradiction, that $a = \{x \mid ord(x)\}$ for a set a. By Lemma 24(4) $\bigcup a = \beta$ is an ordinal. Thus also $\beta^+ \in a$. By the property of the sum operation we know $y \subseteq \bigcup a$ for every $y \in a$. This yields the contradiction $\beta \in \beta^+ \subseteq \beta$.

∎

Exercise 2.11.17 **(1)** Intuitively, this is obvious. The task here is to show that s is finite by Definition 21. By assumption there are natural numbers n_1, n_2 and bijections f_1, f_2 satisfying $func(f_1, n_1, s_1)$ and $func(f_2, n_2, s_2)$. We define

$$f = f_1 \cup \{\langle n_1 + m, y\rangle \mid \langle m, y\rangle \in f_2\}$$

It can be easily seen that f is a bijection and $func(f, n_1 + n_2, s_1 \cup s_2)$ is true.
(2) This is the contraposition of part (1).
(3) Let g be a bijection with $func(g, k, a)$ and f_i bijections with $func(f_i, n_i, g(i))$ for $i < k$. We define

$$f = \bigcup_{i<k} \{\langle \Sigma_{j<i} n_j + m, y\rangle \mid \langle m, y\rangle \in f_i\}$$

Less formal we could write:

$$\begin{aligned} f \;=\; & \{\langle m, y\rangle \mid \langle m, y\rangle \in f_0\} \cup \\ & \{\langle n_0 + m, y\rangle \mid \langle m, y\rangle \in f_1\} \cup \\ & \{\langle n_0 + n_1 + m, y\rangle \mid \langle m, y\rangle \in f_2\} \cup \\ & \ldots \cup \\ & \{\langle \Sigma_{j<k-1} n_j + m, y\rangle \mid \langle m, y\rangle \in f_{k-1}\} \end{aligned}$$

It can be easily seen that f is a bijection and $func(f, \Sigma_{j<k} n_j, \bigcup a)$ is true.
(4) This is again the contraposition of part (3).

∎

Solutions to Chapter 3

Solution to Exercise 3.9.1 The statement of the lemma is rather obvious and one is easily persuaded to believe it. Let us nevertheless present a detailed proof if only for the purpose to see what such a detailed proof would look like.

We show by structural induction on the complexity of a formula $F \in PModFml$

$$\text{for all } g \in G_2$$
$$(G_1, R_1, v_1, g) \models F \text{ iff } (G_2, R_2, v_2, g) \models F.$$

In the initial case the formula F consists of a propositional variable p and we need to prove $v_1(g, p) = v_2(g, p)$. This is immediate from the definition of a subframe.

The proof of the induction step is organized in various cases depending on the leading logical operator of F. The propositional operators are straight forward. We will present the case $\Box F$. The case $\Diamond F$ is absolutely parallel. To save notational overhead we use the abbreviation $\mathcal{K}_i$ for (G_i, R_i, v_i).

Let us first assume $(\mathcal{K}_1, g) \models \Box F$. Thus $(\mathcal{K}_1, h) \models F$ is true for all $h \in G_1$ with $R_1(g, h)$. We need to show $(\mathcal{K}_2, g) \models \Box F$. To this end we consider an arbitrary $h \in G_2$ with $R_2(g, h)$. By the subframe property we also have $R_1(g, h)$ and by what we just said we know $(\mathcal{K}_1, h) \models F$. By induction hypothesis this implies $(\mathcal{K}_2, h) \models F$, as desired.

Now assume for the reverse implication $(\mathcal{K}_2, g) \models \Box F$. Thus $(\mathcal{K}_2, h) \models F$ is true for all $h \in G_2$ with $R_2(g, h)$. We aim to show $(\mathcal{K}_1, g) \models \Box F$. Thus we consider an arbitrary element $h \in G_1$ satisfying $R_1(g, h)$ and need to show $(\mathcal{K}_1, h) \models F$. Since G_2 is a closed subframe of G_1 we get $h \in G_2$ and also $R_2(g, h)$. By the assumption for this part of the proof we obtain $(\mathcal{K}_2, h) \models F$. The induction hypothesis now yields $(\mathcal{K}_2, h) \models F$.

∎

Solution to Exercise 3.9.2

1. We need to exhibit a Kripke structure $\mathcal{K}$ and a world g_1 in $\mathcal{K}$ such that $g_1 \models (\Box P \to \Box Q)$ and $g_1 \not\models \Box(P \to Q)$. See Figure 6.1.

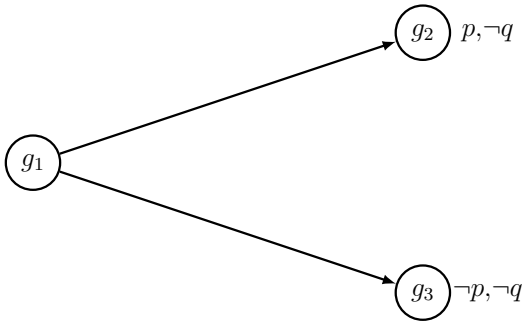


Figure 6.1: Counterexample to $(\Box P \to \Box Q) \to \Box(P \to Q)$

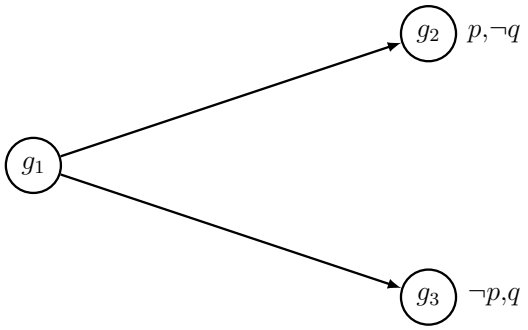


Figure 6.2: Counterexample to $\Box(P \vee Q) \to (\Box P \vee \Box Q)$

2. We need to exhibit a Kripke structure $\mathcal{K}$ and a world g_1 in $\mathcal{K}$ such that $g_1 \models \Box(P \vee Q)$ and $g_1 \not\models (\Box P \vee \Box Q)$. See Figure 6.2.

Solution to Exercise 3.9.3

1. $\Diamond\Box p \leftrightarrow \Box p$
   Let $\mathcal{K} = (G, R, v)$ be a Kripke structure with R an equivalence relation and assume that for an arbitrary $g \in G$ we have $g \models \Diamond\Box p$. Thus there is h_1 with $R(g, h_1)$ such that for all h with $R(h_1, h)$ we know $h \models p$. The description that there is some h_1 such that $R(g, h_1)$ and $R(h_1, h)$ says nothing more than that g and h belong to the same equivalence class of R. More formaly $\exists h_1(R(g, h_1) \wedge R(h_1, h)) \leftrightarrow R(g, h)$. In particular this shows $g \models \Box p$.

2. $\Box\Diamond p \leftrightarrow \Diamond p$

Follows from 1 by the usual trick of replacing p by $\neg p$ and some obvious equivalence transformations.

Solution to Exercise 3.9.4

Proof of 2 Let $\mathcal{K} = (G, R, v)$ be a Kripke structure with (G, R) a symmetric frame and $g \in G$. We want to show $(\mathcal{K}, g) \models p \rightarrow \Box\Diamond p$. So assume $(\mathcal{K}, g) \models p$ and try to prove $(\mathcal{K}, g) \models \Box\Diamond p$. For any g_1 satisfying $R(g, g_1)$ we need to find g_2 with $R(g_1, g_2)$ and $(\mathcal{K}, g_2) \models p$. Symmetry allows us to use $g_2 = g$.

For the proof of the second part of the characterization property consider a frame (G, R) that is not symmetric. There are thus g_1, g_2 with $R(g_1, g_2)$ and $\neg R(g_2, g_1)$. We define a valutation function v as follows:

$$v(h, p) = \begin{cases} 0 & \text{if } R(g_2, h). \\ 1 & \text{otherwise} \end{cases}$$

This definition was specifically taylored to yield $((G, R, v), g_1) \models p$ and $((G, R, v), g_2) \models \neg\Diamond p$. The last formula also yields $((G, R, v), g_1) \models \neg\Box\Diamond p$. Altogether we arrive at $((G, R, v), g_1) \models \neg(p \rightarrow \Box\Diamond p)$.

Proof of 3 Let $\mathcal{K} = (G, R, v)$ be a Kripke structure with (G, R) a serial frame and $g \in G$. We want to show $(\mathcal{K}, g) \models \Box p \rightarrow \Diamond p$. So assume $(\mathcal{K}, g) \models \Box p$ and try to prove $(\mathcal{K}, g) \models \Diamond p$. Since R is serial there is a world g_1 satisfying $R(g, g_1)$. By the assumption $(\mathcal{K}, g) \models \Box p$ we obtain $(\mathcal{K}, g_1) \models p$ and thus $(\mathcal{K}, g) \models \Diamond p$.

For the proof of the second part of the characterization property consider a frame (G, R) that is not serial. There is thus $g_0 \in G$ such that no g_1 exists satisfying $R(g_0, g_1)$. We define a valutation function v with $v(h, p) = 0$ for all $h \in G$. This yields first of all $((G, R, v), g_0) \models \neg\Diamond p$. By the semantics of the Box operator we also have $((G, R, v), g_0) \models \Box p$. Thus $((G, R, v), g_0) \not\models \Box p \rightarrow \Diamond p$, as desired.

Proof of 5 Let $\mathcal{K} = (G, R, v)$ be a Kripke structure with (G, R) a Euclidian frame and $g \in G$. We want to show $(\mathcal{K}, g) \models \Diamond p \rightarrow \Box\Diamond p$. So assume $(\mathcal{K}, g) \models \Diamond p$ and try to prove $(\mathcal{K}, g) \models \Box\Diamond p$. By assumption there exists $g_1 \in G$ with $R(g, g_1)$ and $(\mathcal{K}, g_1) \models p$. For any g_2 with $R(g, g_2)$ we need to find g_3 with $R(g_2, g_3)$ and $(\mathcal{K}, g_3) \models p$. By the Euclidean property we have $R(g_2, g_1)$, and thus $g_3 = g_1$ does the job.

For the proof of the second part of the characterization property consider a frame (G, R) that is not Euclidean. There are g_1, g_2, g_3 with $R(g_1, g_2)$,

$R(g_1, g_3)$ and $\neg R(g_2, g_3)$. We define a valutation function v as follows:

$$v(h,p) = \begin{cases} 0 & \text{if } R(g_2, h). \\ 1 & \text{otherwise} \end{cases}$$

Immediate consequences of this definition are $((G,R,v), g_3) \models p$ and $((G,R,v), g_2) \models \neg\Diamond p$. Because of $R(g_1, g_3)$ we also get $((G,R,v), g_1) \models \Diamond p$. Because of $R(g_1, g_2)$ we also get $((G,R,v), g_1) \models \neg\Box\Diamond p$. Together this shows $((G,R,v), g_1) \not\models \Diamond p \to \Box\Diamond p$.

Proof of 6 Let $\mathcal{K} = (G,R,v)$ be a Kripke structure with (G,R) a weakly functional frame and $g \in G$. We want to show $(\mathcal{K}, g) \models \Diamond p \to \Box p$. So assume $(\mathcal{K}, g) \models \Diamond p$ and try to prove $(\mathcal{K}, g) \models \Box p$. By assumption there is $g_1 \in G$ with $R(g, g_1)$ and $(\mathcal{K}, g_1) \models p$. Consider an arbitrary $g_2 \in G$ with $R(g, g_2)$ we need to prove $(\mathcal{K}, g_2) \models p$. Since weak functionality requires $g_1 = g_2$ this is obvious.

For the proof of the second part of the characterization property consider a frame (G,R) that is not weakly functional. There are thus g, g_1, g_2 satisfying $R(g, g_1)$, $R(g, g_2)$ and $g_1 \neq g_2$. We define a valutation function v as follows:

$$v(h,p) = \begin{cases} 1 & \text{if } h = g_1 \\ 0 & \text{otherwise} \end{cases}$$

Obviously, thus entails $((G,R,v), g_1) \models p$ and $((G,R,v), g_2) \models \neg p$ and further $((G,R,v), g) \models \Diamond p$ and $((G,R,v), g) \models \neg\Box p$. Altogether $((G,R,v), g) \not\models \Diamond p \to \Box p$ and we are done.

Proof of 7 Follows immediately from parts 3 and 6.

Proof of 8 Let $\mathcal{K} = (G,R,v)$ be a Kripke structure with (G,R) a dense frame and $g \in G$. We want to show $(\mathcal{K}, g) \models \Box\Box p \to \Box p$. So assume $(\mathcal{K}, g) \models \Box\Box p$ and try to prove $(\mathcal{K}, g) \models \Box p$. We thus consider $g_2 \in G$ satisfying $R(g, g_2)$ and aim to show $(\mathcal{K}, g_2) \models p$. Since R is dense we are sure to find g_1 with $R(g, g_1)$ and $R(g_1, g_2)$. But, now we can use $(\mathcal{K}, g) \models \Box\Box p$ to conclude $(\mathcal{K}, g_2) \models p$.

For the proof of the second part of the characterization property consider a frame (G,R) that is not dense. This means that there exist g_0, g_1 in G, such that there is no $g \in G$ *between* them i.e., there is no $g \in G$ with $R(g_0, g)$ and $R(g, g_1)$. We define a valutation function v as follows:

$$v(h,p) = \begin{cases} 1 & \text{if } g \in G \text{ exists with } R(g_0, g) \text{ and } R(g, h). \\ 0 & \text{sonst} \end{cases}$$

Obviously $((G,R,v),g_0) \models \Box\Box p$ and $((G,R,v),g_1) \models \neg p$. Thus also $((G,R,v),g_0) \models \neg\Box p$. This shows that $((G,R,v),g_0) \models \Box\Box p \rightarrow \Box p$ is not true and we are done.

Proof of 9 Let $\mathcal{K} = (G,R,v)$ be a Kripke structure with (G,R) a weakly connective frame and $g \in G$. We want to show $(\mathcal{K},g) \models \Box((p \wedge \Box p) \rightarrow q) \vee \Box((q \wedge \Box q) \rightarrow p)$. We assume that the first part of the disjunction is not true, i.e., that there is $g_1 \in G$ with $R(g,g_1)$, $(\mathcal{K},g_1) \models p \wedge \Box p$ and $(\mathcal{K},g_1) \models \neg q.$, and we will show that the second disjunct has to be true. To this end consider an arbitrary g_2 with $R(g,g_2)$ and $(\mathcal{K},g_2) \models (q \wedge \Box q)$ and try to show $(\mathcal{K},g_2) \models p$. By weak connectivity there are three possibilities: (a) $g_1 = g_2$ (b) $R(g_2,g_1)$ and (c) $R(g_1,g_2)$. Alternative (a) yields a contradiction since we have $(\mathcal{K},g_2) \models q$ and $(\mathcal{K},g_1) \models \neg q$. Also alternative (b) is contradictory since we have $(\mathcal{K},g_2) \models \Box q$ and $(\mathcal{K},g_1) \models \neg q$. Thus alternative (c) has to be the case. But now $(\mathcal{K},g_1) \models \Box p$ yields $(\mathcal{K},g_2) \models p$ as desired.

For the proof of the second part of the characterization property consider a frame (G,R) that is not weakly connective. Thus there are $s,s_1,t_1 \in G$ with $R(s,s_1)$ and $R(s,t_1)$ but

$$\neg R(s_1,t_1) \quad \text{and} \quad \neg R(t_1,s_1) \quad \text{and} \quad t_1 \neq s_1$$

We define a interpretation v by:

$$v(h,p) = \begin{cases} 1 & \text{if } h = s_1 \text{ or } R(s_1,h) \\ 0 & \text{otherwise} \end{cases}$$

$$v(h,q) = \begin{cases} 1 & \text{if } h = t_1 \text{ or } R(t_1,h) \\ 0 & \text{otherwise} \end{cases}$$

This stipulation yields in particular $v(t_1,p) = 0$ and $v(s_1,q) = 0$. It is furthermore easy to see that $s_1 \models p \wedge \Box p$ holds true. Combined these observations yield $s_1 \not\models (p \wedge \Box p) \rightarrow q$ and also $s \not\models \Box(p \wedge \Box p) \rightarrow q$. Analogously $s \not\models \Box(q \wedge \Box q) \rightarrow p$ can be seen to be true. Altogether we arrive at

$$((G,R,v),s) \not\models \Box((p \wedge \Box p) \rightarrow q) \vee \Box((q \wedge \Box q) \rightarrow p)$$

and are done.

Proof of 10 Let $\mathcal{K} = (G,R,v)$ be a Kripke structure with (G,R) a weakly oriented frame and $g \in G$. We want to show $(\mathcal{K},g) \models \Diamond\Box p \rightarrow \Box\Diamond p$. So assume $(\mathcal{K},g) \models \Diamond\Box p$ and try to prove $(\mathcal{K},g) \models \Box\Diamond p$. There is thus g_1

with $R(g, g_1)$ and $(\mathcal{K}, g_1) \models \Box p$. To meet our proof obligation we consider an arbitrary g_2 with $R(g, g_2)$ and need to find g_3 with $R(g_2, g_3)$ and $(\mathcal{K}, g_3) \models p$. Since R is weakly oriented we find $h \in G$ with $R(g_1, h)$ and $R(g_2, h)$. From $(\mathcal{K}, g_1) \models \Box p$ we get $(\mathcal{K}, h) \models p$. This shows that we can use $g_3 = h$.

For the proof of the second part of the characterization property consider a frame (G, R) that is not weakly oriented. Thus we find g, g_1, g_2 in G with $R(g, g_1)$ and $R(g, g_2)$ such that there is no h satisfying simultaneously $R(g_1, h)$ and $R(g_2, h)$. We define a interpretation v by:

$$v(h, p) = \begin{cases} 1 & \text{if } R(g_1, h) \\ 0 & \text{otherwise} \end{cases}$$

This definition gives immediately $((G, R, v), g_1) \models \Box p$ and $((G, R, v), g) \models \Diamond\Box p$. The definition was furthermore taylored to yield $((G, R, v), g_2) \models \neg\Diamond p$. Thus entails $((G, R, v), g) \models \neg\Box\Diamond p$. In summa $((G, R, v), g) \not\models \Diamond\Box p \rightarrow \Box\Diamond p$.

Proof of 11 Let $\mathcal{K} = (G, R, v)$ be a Kripke structure with (G, R) a confluent frame and $g \in G$. We want to show $(\mathcal{K}, g) \models \Diamond\Box p \rightarrow \Diamond p$. So assume $(\mathcal{K}, g) \models \Diamond\Box p$ and try to prove $(\mathcal{K}, g) \models \Diamond p$. By this assumption there exists g_1 with $R(g, g_1)$ and $(\mathcal{K}, g_1) \models \Box p$. We want to find g_2 with $R(g, g_2)$ and $(\mathcal{K}, g_2) \models p$. By the confluence property there is h with $R(g, h)$ and $R(g_1, h)$. Thus $(\mathcal{K}, g_1) \models \Box p$ implies $(\mathcal{K}, h) \models p$ which shows that $g_2 = h$ is possible.

For the proof of the second part of the characterization property consider a frame (G, R) that is not confluent. There are thus elements g_1, g_2 in G with $R(g_1, g_2)$ and there is no h satisfying both $R(g_1, h)$ and $R(g_2, h)$. We define a interpretation v by:

$$v(h, p) = \begin{cases} 1 & \text{if } R(g_2, h) \\ 0 & \text{otherwise} \end{cases}$$

The definition immediately implies $((G, R, v), g_2) \models \Box p$ and thus also $((G, R, v), g_1) \models \Diamond\Box p$. On the other hand $((G, R, v), g_1) \models \Diamond p$ cannot be true, since this would entail the existence of a world h with $R(g_1, h)$ and $((G, R, v), h) \models p$. By definition of v the last claim would imply $R(g_2, h)$ which is excluded by our assumptions. Thus we arrive at $((G, R, v), g_1) \not\models \Diamond\Box p \rightarrow \Diamond p$.

Solution to Exercise 3.9.5

1. By definition $C(0,1,2,0)$ stands for

$$\forall w_1 \forall w_2 \forall w_3((w_1 = w_2 \wedge R^2(w_1, w_3)) \rightarrow \exists w_4(R(w_2, w_4) \wedge w_3 = w_4))$$

We may replace w_2 everywhere by w_1 and w_4 everywhere by w_3 to obtain the following logically equivalent formula

$$\forall w_1 \forall w_3(R^2(w_1, w_3) \rightarrow R(w_1, w_3)),$$

If we expand $R^2(w_1, w_3)$ to its definition we are looking at the familiar definition for transitivity of R.

2. Unravelling the definition for $C(0,1,0,0)$ we get

$$\forall w_1 \forall w_2 \forall w_3((w_1 = w_2 \wedge w_1 = w_3) \rightarrow \exists w_4(R(w_2, w_4) \wedge w_3 = w_4))$$

Replacing both w_2 and w_3 everywhere by w_1 we arrive after some trivial transformations at

$$\forall w_1 \exists w_4(R(w_1, w_4) \wedge w_1 = w_4)$$

We can omit existential quantification by replacing w_4 by w_1 and obtain after omitting the trivially true conjunct $w_1 = w_1$

$$\forall w_1 R(w_1, w_1)$$

3. Replacing $C(1,1,0,0)$ by its definition we obtain

$$\forall w_1 \forall w_2 \forall w_3((R(w_1, w_2) \wedge w_1 = w_3) \rightarrow \exists w_4(R(w_2, w_4) \wedge w_3 = w_4))$$

Replacing both w_3 and w_4 by w_1 which is an operation preserving logical equivalence we get after some trivial transformations

$$\forall w_1 \forall w_2(R(w_1, w_2) \rightarrow R(w_2, w_1))$$

4. Unfolding the definition of $C(1,0,1,0)$ yields

$$\forall w_1 \forall w_2 \forall w_3(R(w_1, w_2) \wedge R(w_1, w_3)) \rightarrow \exists w_4(w_2 = w_4 \wedge w_3 = w_4))$$

Here it suffices to replace w_4 by w_3 ain the desired result

$$\forall w_1 \forall w_2 \forall w_3(R(w_1, w_2) \wedge R(w_1, w_3) \rightarrow w_2 = w_3)$$

See also [Sterling, 1992, page 493].

Solution to Exercise 3.9.6 Both equivalences are proved by induction on n. We will only present the proof of the first statement, the second proof proceeds analogously.

For $n = 0$ the claim is reduced to

$$(\mathcal{K}, g) \models F \text{ iff for all } h \in G \text{ with } R^0(g, h) \text{ we have } (\mathcal{K}, h) \models F.$$

Since $R^0(g, h) = g \doteq h$ this is further equivalent to

$$(\mathcal{K}, g) \models F \text{ iff } (\mathcal{K}, g) \models F.$$

Proof of the induction step from n to $n + 1$:

$$\begin{array}{lll} (\mathcal{K}, g) \models \Box^{n+1} F & \text{iff} & (\mathcal{K}, h_1) \models \Box^n F \text{ for all } h_1 \text{ with } R(g, h_1) \quad \text{Def. of } \Box^{n+1} \\ & \text{iff} & (\mathcal{K}, h) \models F \text{ for all } h, h_1 \text{ with} \\ & & R(g, h_1) \text{ and } R^n(h_1, h) \quad \text{Ind.Hyp.} \\ & \text{iff} & (\mathcal{K}, h) \models F \text{ for all } h \text{ with } R^{n+1}(g, h) \quad \text{Def. of } R^{n+1} \end{array}$$

Solution to Exercise 3.9.7 Assume by way of contradiction that there is a formula $F \in PModFml$ characterizing the class $\{(G, R) \mid (G, R) \models \exists x \exists y R(x, y)\}$. We consider two frames (G_1, R_1) and (G_2, R_2) with $G_1 = \{g_1, g_2, g_3\}$ and $G_2 = \{g_1\}$. Furthermore $R_1(x, y) \Leftrightarrow x = g_2$ and $y = g_3$ and finally R_2 the empty relation. By our assumptions on F there is a valuation v_2 with $((G_2, R_2, v_2), g_1) \models \neg F$ and $((G_1, R_1, v), g_i) \models F$ for all v and all $i \in \{1, 2, 3\}$. We choose some v_1 such that v_2 is the restriction of v_1 to G_2. Obviously (G_2, R_2) is a closed subframe of (G_1, R_1). Thus by the result from Exercise 3.9.1 $((G_1, R_1, v_1), g_1) \models F$ implies $((G_2, R_2, v_2), g_1) \models F$.
This contradiction completes the proof.

Solution to Exercise 3.9.8 Let $\mathcal{K} = (G, R, v)$ be the disjoint sum of the Kripke structures $\mathcal{K}_1 = (G_1, R_1, v_1)$ and $\mathcal{K}_2 = (G_2, R_2, v_2)$. For simplicity we assume that $G_1 \cap G_2 = \emptyset$.

We set out ot show $\quad \mathcal{K} \models F$ iff $\mathcal{K}_1 \models F$ and $\mathcal{K}_2 \models F$

To this end we show by induction on the complexity of F and every world $g \in G$

$$(\mathcal{K}, g) \models F \text{ iff } (\mathcal{K}_1, g) \models F \text{ or } (\mathcal{K}_2, g) \models F$$

Since $G = G_1 \uplus G_2$ it suffices to prove

1. for every $g \in G_1$: $(\mathcal{K}, g) \models F$ iff $(\mathcal{K}_1, g) \models F$
2. for every $g \in G_2$: $(\mathcal{K}, g) \models F$ iff $(\mathcal{K}_2, g) \models F$

In the base case of the induction F is a propositional variable and the claims follow immediately from the definition of v.

We will present the induction step for the case $F = \Box F_0$. The remaining cases are either simple of analogous.

We observe that the definition of R yields for all $g \in G_i$ and $h \in G$ $R(g,h) \Leftrightarrow h \in G_i \wedge R_i(g,h)$. This observation an the induction hypothesis immediately lead to

1. for every $g \in G_1$: $(\mathcal{K}, g) \models \Box F_0$ iff $(\mathcal{K}_1, g) \models \Box F_0$
2. for every $g \in G_2$: $(\mathcal{K}, g) \models \Box F_0$ iff $(\mathcal{K}_2, g) \models \Box F_0$

as required.

Solution to Exercise 3.9.9 Consider the two frames (G_1, R_1), (G_2, R_2) with $G_1 = \{g_1\}$, $G_2 = \{g_2\}$, $g_1 \neq g_2$ and $R_i(g_i, g_i)$ for $i = 1, 2$. Assume that F characterizes the class of all universal frames. Thus for arbitrary interpretations w_i we have $(G_i, R_i, w_i) \models F$. By choice of F we also know that there is a interpretation v such that $(G, R, v) \models \neg F$ for the disjoint sum $(G, R) = (G_1, R_1) \uplus (G_2, R_2)$. Let v_1, v_2 be such that $(G, R, v) = (G_1, R_1, v_1) \uplus (G_2, R_2, v_2)$. Note, that v_1, v_2 can easily be found since G_1 and G_2 are disjoint. From $(G_i, R_i, v_i) \models F$ we conclude by Exercise 3.9.8 also $(G, R, v) \models F$. Contradiction.

Solution to Exercise 3.9.10

1	F_{trans}	$\forall X(\forall u(R(z,u) \to X(u)) \to \forall w \forall v(R(z,w) \wedge R(w,v) \to X(v)))$
2	$\neg F_{trans}$	$\exists X(\forall u(R(z,u) \to X(u)) \wedge \exists w \exists v(R(z,w) \wedge R(w,v) \wedge \neg X(v)))$

Apply lemma with

$G \equiv R(z,u)$

$H \equiv \exists w \exists v(R(z,w) \wedge R(w,v) \wedge \neg X(v))$ negative in X

3	$\neg F_{trans}$	$\exists w \exists v(R(z,w) \wedge R(w,v) \wedge \neg R(z,v))$
4	F_{trans}	$\forall w \forall v(R(z,w) \wedge R(w,v) \to R(z,v))$

Solution to Exercise 3.9.11

1	F_{ser}	$\forall X(\forall x(R(z,x) \to X(x)) \to \exists u(R(z,u) \wedge X(u)))$
2	$\neg F_{ser}$	$\exists X(\forall x(R(z,x) \to X(x)) \wedge \forall u(R(z,u) \to \neg X(u)))$

Apply lemma with

$G \equiv R(z,x)$

$H \equiv \forall u(R(z,u) \to \neg X(u))$ negative in X

3	$\neg F_{ser}$	$\forall u(R(z,u) \to \neg R(z,u))$
4	$\neg F_{ser}$	$\forall u(\neg R(z,u))$
5	F_{ser}	$\exists u R(z,u)$
6	$\forall z F_{ser}$	$\forall z \exists u R(z,u)$

Solution to Exercise 3.9.12

The characterizing formula is $\Box$**false**.
For a Kripke structure $\mathcal{K} = (G, R, v)$ with $R = \emptyset$ and arbitrary $g \in G$ the claim $g \models \Box$**false** is true of for all $h \in G$ with $R(g,h)$ we have $h \models$ **false**. This is vacuously true since there is no h with $R(g,h)$. Also the reverse implication is simple. If, for the sake of a contradiction, there would be $g, h \in G$ with $R(g,h)$ then $g \models \Box$**false** could not be true for arbitrary interpretation v.

Solution to Exercise 3.9.13

Fix n. The proof proceed by induction on $n-k$, or reverse induction on k from $k = n$ to $k = 0$
If $n - k = 0$, i.e. $k = n$, then F is a purely propositional formula without modal operators. So, $(\mathcal{K}, g_1) \models F$ only depends on $v(g_1, p)$ for $p \in$ PVar. But, $v(g_1, p) = v^n_g(g_1, p)$ for any n and any g with $g_1 \in G^n_g$.
In the inductive step from $k+1$ to k we assume that for any F' with $md(F') \leq n - k - 1$ and $g' \in G^{k+1}_g$ we have

$$(\mathcal{K}, g') \models F' \Leftrightarrow (\mathcal{K}^n_g, g') \models F'$$

and we try to establish that for any F with $md(F) \leq n - k$ and any $g_1 \in G^k_g$

$$(\mathcal{K}, g_1) \models F \Leftrightarrow (\mathcal{K}^n_g, g_1) \models F$$

This inductive step is proved by structural induction on F. The case that F is a propositional variable has already be dealt with. If $F = F_1 \wedge F_2$ then also $md(F_i) \leq n - k$. By the semantics definition $(\mathcal{K}, g_1) \models F$ is equivalent

to $(\mathcal{K}, g_1) \models F_1$ and $(\mathcal{K}, g_1) \models F_1$. By the induction hypothesis of the structural induction this is equivalent to $(\mathcal{K}^n_g, g_1) \models F_1$ and $(\mathcal{K}^n_g, g_1) \models F_1$ which again by the semantics definition is equivalent to $(\mathcal{K}^n_g, g_1) \models F$. The other propositional cases $F = F_1 \vee F_2$ and $F = \neg F_1$ follow analogously.

So let us look at $F = \Box F_1$ with $md(F) \leq n - k$. Thus $md(F_1) \leq n - k - 1$.

Assume $(\mathcal{K}, g_1) \models \Box F_1$ with the aim to arrive at $(\mathcal{K}^n_g, g_1) \models F$. To achieve this aim we have to prove for any $g_2 \in G^n_g$ with $R^n_g(g_1, g_2)$ that $(\mathcal{K}^n_g, g_2) \models F_1$. We first point out that under the present assumptions $R^n_g(g_1, g_2)$ is equivalent to $R(g_1, g_2)$. This gives us $(\mathcal{K}, g_2) \models F_1$. Since $g_1 \in G^k_g$ implies $g_2 \in G^{k+1}_g$ the outer induction hypothesis is applicable and yields $(\mathcal{K}^n_g, g_2) \models F_1$ as desired.

Now assume $(\mathcal{K}^n_g, g_1) \models F$ with the aim to arrive at $(\mathcal{K}, g_1) \models \Box F_1$. To achieve this aim we have to prove for any $g_2 \in G^n_g$ with $R(g_1, g_2)$ that $(\mathcal{K}, g_2) \models F_1$. Again $R(g_1, g_2)$ is equivalent to $R^n_g(g_1, g_2)$ which gives us $(\mathcal{K}^n_g, g_2) \models F_1$. Since $g_1 \in G^k_g$ implies $g_2 \in G^{k+1}_g$ the outer induction hypothesis is applicable and yields $(\mathcal{K}, g_2) \models F_1$ as desired.

The case $F = \Diamond F_1$ follows analogously.

Solution to Exercise 3.9.14

We start with the easy implication. Assume $\vdash (F_1 \wedge \Box F_1 \wedge \ldots \Box^n F_1) \to F_2$ and try to prove $F_1 \vdash_G F_2$. So we look at a Kripke structure $\mathcal{K} = (G, R, v)$ with $(\mathcal{K}, g) \models F_1$ for all $g \in G$ and want to arrive at $(\mathcal{K}, g_1) \models F_2$ for any $g_1 \in G$. Since $(\mathcal{K}, g) \models F_1$ is true for all $g \in G$ we have $(\mathcal{K}, g_1) \models F_1 \wedge \Box F_1 \wedge \ldots \Box^n F_1$. Thus by assumption $(\mathcal{K}, g_1) \models F_2$, as desired.

For the reverse implication we assume $F_1 \vdash_G F_2$ and for an arbitrary Kripke structure $\mathcal{K} = (G, R, V)$ we assume $(\mathcal{K}, g_1) \models F_1 \wedge \Box F_1 \wedge \ldots \Box^n F_1$ for an arbitrary $g_1 \in G$ with the aim to show $(\mathcal{K}, g_1) \models F_2$. Let $K^n_{g_1} = (G^n_{g_1}, R^n_{g_1}, v^n_{g_1})$ be the Kripke structure defined in Exercise 3.9.13. From the definition of $G^n_{g_1}$ we see that $(\mathcal{K}, g_1) \models F_1 \wedge \Box F_1 \wedge \ldots \Box^n F_1$ implies $(\mathcal{K}, g) \models F_1$ for all $g \in G^n_{g_1}$. Since $md(F_1 \leq n$ we get from Excercise 3.9.13 also $(\mathcal{K}^n_{g_1}, g) \models F_1$ for all $g \in G^n_{g_1}$. The assumption $F_1 \vdash_G F_2$ now yields $(\mathcal{K}^n_{g_1}, g) \models F_2$ for all $g \in G^n_{g_1}$, in particular $(\mathcal{K}^n_{g_1}, g_1) \models F_2$. Since $md(F_1 \leq n$ another appeal to Excercise 3.9.13 yields $(\mathcal{K}, g_1) \models F_2$ and we have finished.

Solution to Exercise 3.9.15

ad 1 From the description logic vocabulary $\mathbf{V} = \mathbf{C} \cup \mathbf{R}$ we construct a first-order vocabulary Σ that contains a unary predicate symbol $C(x)$ for each $C \in \mathbf{C}$ and a binary relation symbol $R(x, y)$ for each $R \in \mathbf{R}$. We use the same symbols to denote the concept and likewise the unary relation, and the role and the binary relation in the hope that this will not cause any confusion.
For an expression C we define its first-order translation C^* inductively as follows:

concept expression C	C^*	comment
C	$C(x)$	if $C \in \mathbf{C}$
$\neg c$	$\neg C^*$	
$C_1 \sqcap C_2$	$C_1^* \wedge C_2^*$	
$C_1 \sqcup C_2$	$C_1^* \vee C_2^*$	
$\forall R.C$	$\forall y(R(x, y) \to C^*(y/x))$	
$\exists R.C$	$\exists y(R(x, y) \wedge C^*(y/x))$	

We arrange this translation in a way that C^* contains x as the only free variable. $C^*(y/x)$ denotes that formula obtained by replacing all occurences of the free variable x by y.
ad 2 `needs to be done`

Solution to Exercise 3.9.16

We need to show for any interpretation $\mathcal{I}$ that $(\neg\forall R.C)^{\mathcal{I}}$ equals $(\exists R.\neg C)^{\mathcal{I}}$. By the semantics definition $(\forall R.C)^{\mathcal{I}}$ is the set $\{d \in \Delta \mid$ for all b with $R(d, b)$ we have $b \in C^{\mathcal{I}}\}$. So its complement is $\{d \in \Delta \mid$ there exists b with $R(d, b)$ and $b \in (\neg C)^{\mathcal{I}}\}$.

Solution to Exercise 3.9.17

That is easy. Transitivity of r says $\forall x\forall y\forall z(r(x, y) \wedge r(y, z) \to r(x, z))$ and transitivity of r^- says $\forall x\forall y\forall z(r^-(x, y) \wedge r^-(y, z) \to r^-(x, z))$. By definition of r^- this is equivalent to $\forall x\forall y\forall z(r(y, x) \wedge r(z, y) \to r(z, x))$ which is just a simple permutation of the definition of transitivity for r.

Solution to Exercise 3.9.18

The answer is No. The formulas in $\mathcal{H}$ written as first-order formulas are

all univeral Horn clauses, i.e. formulas of the form $\forall \bar{x}(lhs \rightarrow rhs)$. These formulas can trivially be satisfied by an interpretation $\mathcal{I}$ with $R^{\mathcal{I}} = \Delta$ for all $R \in \mathbf{R}$.
By the way $\mathcal{I}$ satisfies $R_1 \sqsubseteq R_2^-$ and $R_2 \sqsubseteq R_1$ iff $R_1^{\mathcal{I}} = R_2^{\mathcal{I}}$ and $R_1^{\mathcal{I}}$ is symmetric.

Solution to Exercise 3.9.19

Let $C \in \mathbf{C}_n$ and $C = C_1$ the defining equation for C in the T-Box $\mathcal{T}$. We built a tree T whose nodes will be labeled by name concepts. The root is labeled with C. In general if a node in the tree is labeled by D from $\mathbf{C}_n$ then there are successor nodes one for each D_i from $\mathbf{C}_n$ that is directly used by D. Since $\mathcal{T}$ is noncyclic T is a finite tree. Now perform the following construction: replace every name symbol in C_1 by its definition to obtain C_2. If C_2 does not contain any name symbol we stop. Otherwise we again replace all name symbols in C_2 by their definition to obtain C_3. And so on to obtain C_k. Any name symbol C' occuring in C_k is on a branch of the tree T. Thus the described replacement process stops after finitely many iterations and we obtain an equation $C = C_k$, where C_k only contains base symbols. Obviously, the original T-Box $\mathcal{T}$ and the one, call it $\mathcal{T}'$, obtained by replacing $C = C_1$ by $C = C_k$ are equivalent, i.e., are satisfied by the same interpretations. But in $\mathcal{T}'$ C is uniquely defined by base concepts.

Solution to Exercise 3.9.20

The claim $\mathcal{K}, g* \models F \quad \Leftrightarrow \quad \mathcal{K}^*, g* \models F$ is proved by structural induction on F.

Solution to Exercise 3.9.21

Let us first look at the easy direction. Assume $\{\Box^n A \mid n \geq 0\} \models_L F$ and try to prove $A \models_G F$. We consider a Kripke structure $\mathcal{K} = (G, R, v)$ with $g \models A$ for all $g \in G$. We want to prove $h \models F$ for all $h \in G$. From the assumption we get $h \models \Box^n A$ for all $n \geq 0$. Which by case assumption yields $h \models F$.
Now assume $A \models_G F$ and consider a Kripke structure $\mathcal{K} = (G, R, v)$ with $g \in G$ with $\mathcal{K}, g \models \Box^n A$ for all $n \geq 0$. We want to arrive at $\mathcal{K}, g \models F$. Let $\mathcal{K}^* = (G^*, R^*, v^*)$ be the Kripke structure from Exercise 3.9.20. Then we have also have $\mathcal{K}^*, g \models \Box^n A$ for all $n \geq 0$. Since all worlds in $\mathcal{K}^*$ are

reachable from g we obtain $\mathcal{K}^*, h \models A$ for all $h \in G^*$. By $A \models_G F$ this implies $\mathcal{K}^*, h \models F$ for all $h \in G^*$. In particular $\mathcal{K}^*, g \models F$. By Exercise 3.9.20 this implies $\mathcal{K}, g \models F$.

∎

Solution to Exercise 3.9.22

`to be done`

Solution to Exercise 3.9.23

⟨z1, line of, 1982953 ⟩	⟨z2, line of, 1982953 ⟩	⟨z3, line of, 1982953 ⟩
⟨z1, item, Fleece⟩	⟨z1, Id, 313169M6⟩	⟨z1, color, orange⟩
	⟨z1, size, M⟩	⟨z1, qu, 2⟩
⟨z2, item, Shirt⟩	⟨z2, Id, 1911409M6⟩	⟨z2, color, smaragd⟩
	⟨z2, size, L⟩	⟨z2, qu, 1⟩
⟨z3, item, Top⟩	⟨z3, Id, 249708M6⟩	⟨z3, color, navy dotted⟩
	⟨z3, size, S⟩	⟨z3, qu, 1⟩

Solution to Exercise 3.9.24

1. $\forall r^{-1}.(C \sqcap \neg\exists s.D) \Leftrightarrow \forall y(r^{-1}(x,y) \rightarrow (C(y) \wedge \neg\exists z(s(y,z) \wedge D(z)))$
   $\Leftrightarrow \forall y(r(y,x) \rightarrow (C(y) \wedge \neg\exists z(s(y,z) \wedge D(z)))$
   $\Leftrightarrow \forall y(r(y,x) \rightarrow (C(y) \wedge \forall z(s(y,z) \rightarrow \neg D(z)))$
2. `to be done`
3. `to be done`

Solution to Exercise 3.9.25

In the proof of the correctness theorem, Theorem 69, the statement that the initial tableau is satisfiable is used. This involves a mapping I of one prefix σ into the set of possible worlds of an arbitrary Kripke structure with reflexive accessability relation such that $R(I(\sigma), I(\sigma))$ if σ is $\mathbf{T}$-accessible from σ. Since σ is $\mathbf{T}$-accessible from σ it is crucial that R is indeed reflexive.

Furthermore the correctness theorem builds on the validity of the correctness lemma. So, let us look at the proof of Lemma 70. The case of the ν-rule (box rule) is valid regardless of the accessibility of prefixes, since the mapping I is not changed. But, the argument concerning the μ-rule (diamond rule) in generel needs reconsideration. But, in the special case of **T**-accessible the text as it stand remains a valid proof.

The critical part of the completeness theorem is the construction of the countermodel (G, R, v) with G the set of prefixes occuring on the open branch and R the accessibility relation. We have to convince ourselves that **T**-accessible entails reflexivity of R. But this is obvious.

Solution to Exercise 3.9.26

Given $I(Tr)$ define

$$ext(d) = \{(d_1, d_2) \mid (d_1, d, d_2) \in I(Tr)\}$$

Given ext define

$$I(Tr) = \{(d_1, d, d_2) \mid (d_1, d_2) \in ext(d)\}$$

Solution to Exercise 3.9.27

Both inclusions are not universally valid. Here are minimal counterexamples $\mathcal{I}_1$ and $\mathcal{I}_2$ with the common universe $\Delta = \{e, d\}$ and $R^{\mathcal{I}_1} = R^{\mathcal{I}_2} = \{(e, d)\}$ and
1 $C_1^{\mathcal{I}_1} = \{e\}$, $C_2^{\mathcal{I}_1} = \{d\}$
Thus $(C_1 \sqcap \exists R.C_2)^{\mathcal{I}_1} = \{e\}$ and $(\exists R.(C_1 \sqcap C_2))^{\mathcal{I}_1} = \emptyset$.
2 $C_1^{\mathcal{I}_1} = C_2^{\mathcal{I}_1} = \{d\}$
Thus $(\exists R.(C_1 \sqcap C_2))^{\mathcal{I}_1} = \{e\}$ and $(C_1 \sqcap \exists R.C_2)^{\mathcal{I}_1} = \emptyset$.

∎

Consider the translation $C_1(x) \wedge \exists y(R(x, y) \wedge C_2(y))$ of $C_1 \sqcap \exists R.C_2$ into first-order logic. This formula is equivalent to its prenex normal form $\exists y(C_1(x) \wedge R(x, y) \wedge C_2(y))$. This is possible since quantified object are explicitely named. In description logic there is no prenex normal form.

Solutions to Chapter 4

Solution to Exercise 4.7.1

The proof proceeds, of course, by induction on the structural complexity of F.

If $F = p$ for a propositional variable $p \in \mathrm{PVar}$ the claim reduces to:

$$g \models p \quad \Leftrightarrow \quad v(g,p) = 1$$

which is true by definition of v. Since the propositional induction steps are trivial it remains by the restrictions placed on F to consider $F = [a]F_1$ (respectively $F = \langle a \rangle F_1$).

$$\begin{array}{lcll} (\mathcal{K},g) \models [a]F_1 & \Leftrightarrow & (\mathcal{K},g') \models F_1 \text{ for all } (g,g') \in \rho(a) & \text{semantics} \\ & \Leftrightarrow & (\mathcal{K}^a,g') \models F_1^a \text{ for all } (g,g') \in \rho(a) & \text{ind.hyp.} \\ & \Leftrightarrow & (\mathcal{K}^a,g') \models F_1^a \text{ for all } R(g,g') & \text{def.of } R \\ & \Leftrightarrow & (\mathcal{K}^a,g) \models \Box F_1^a & \text{semantics} \\ & \Leftrightarrow & (\mathcal{K}^a,g) \models F^a & \text{def.of } F^a \end{array}$$

The case $F = \langle a \rangle F_1$ follows analogously.

Solution to Exercise 4.7.2

$(u,w) \in \rho(\alpha;(\neg A?;\alpha)^*;A?)$
iff there exist $v \in S$ such that $(u,v) \in \rho(\alpha)$ and $(v,w) \in \rho((\neg A?;\alpha)^*;A?)$
iff there exist $n \in \mathbb{N}$ and $u_1,\ldots,u_n \in S$ with $u_1 = v$, $u_n = w$
$(u,v) \in \rho(\alpha)$ and
$u_i \models \neg A$ and $(u_i,u_{i+1}) \in \rho(\alpha)$ for all $1 \leq i < n$ and
$w \models A$
iff **repeat** α **until** A

Solution to Exercise 4.7.3

Set $A? \equiv$ **if** A **then** α_{skip} **else** α_{nt} where α_{skip} is a program that always terminates without state change and α_{nt} is non-terminating progam. More precisely for any Kripke structure $(S,\rho,\models)$ we have

$$\begin{array}{lcl} \rho(\alpha_{skip}) & = & \{(s,s) \mid s \in S\} \\ \rho(\alpha_{nt}) & = & \emptyset \end{array}$$

Then

$$\begin{aligned}\rho(\textbf{if } A \textbf{ then } \alpha_{skip} \textbf{ else } \alpha_{nt}) &= \{(s_1,s_2) \mid \ (s_1,s_2) \in \rho(\alpha_{skip}) \text{ if } s_1 \models A \\ &\qquad\qquad\quad\ (s_1,s_2) \in \rho(\alpha_{nt}) \text{ if } s_1 \models \neg A\} \\ &= \{(s_1,s_2) \mid \ (s_1,s_2) \in \rho(\alpha_{skip}) \text{ if } s_1 \models A\} \\ &= \{(s_1,s_1) \mid \ \text{if } s_1 \models A\} \\ &= \rho(A?)\end{aligned}$$

Solution to Exercise 4.7.4

Let $\mathcal{K}$ be a Kripke structure and s one of its states.
Proof of (1). We need to show $s \models \neg\langle p\rangle F \leftrightarrow [p]\neg F)$

$$\begin{array}{lll} s \models \neg\langle\pi\rangle F & \text{iff} & \text{there is no } t \text{ with } (s,t) \in \rho(\pi) \text{ and } t \models F \\ & \text{iff} & \text{for all } t \text{ satisfying } (s,t) \in \rho(\pi) \text{ we have } t \models \neg F \\ & \text{iff} & s \models [p]\neg F \end{array}$$

Proof of (2). We need to show $s \models \neg[\pi]F \leftrightarrow \langle\pi\rangle\neg F)$

$$\begin{array}{lll} s \models \neg[\pi]F & \text{iff} & \text{it is not true that for all } (s,t) \in \rho(\pi) \text{ we have } t \models F \\ & \text{iff} & \text{there is at least one } t \text{ with } (s,t) \in \rho(\pi) \text{ and } t \models \neg F \\ & \text{iff} & s \models \langle\pi\rangle\neg F \end{array}$$

Proof of (3). We need to show $s \models [\pi](F \to G) \to (([\pi]F) \to [\pi]G)$

$$s \models [\pi](F \to G) \quad \text{iff} \quad \text{for all } t \text{ with } (s,t) \in \rho(\pi) \text{ we have } t \models (F \to G)$$

Now, assume $s \models [\pi]F$, i.e., for all t such that $(s,t) \in \rho(\pi)$ we get $t \models F$. Since also $t \models (F \to G)$ is true we conclude $t \models G$. This gives for all t with $(s,t) \in \rho(\pi)$ the statement $t \models G$, i.e., $s \models [\pi]G$. Altogether $s \models (([\pi]F) \to ([\pi]G))$.

Solution to Exercise 4.7.5

Assume $s_1 \models \neg\langle\pi\rangle F$ implies $s_2 \models \neg F$ and $s_1 \models [\pi]\neg F$. We want to show $s_2 \models \neg F$. From $s_1 \models [\pi]\neg F$ we get by duality of the modal operators $s_1 \models \neg\langle\pi\rangle F$. This immediately yields $s_2 \models \neg F$.

Solution to Exercise 4.7.6

If $\vdash G \to F$ then obviously $G \vdash F$. This is the easy implication that would

also be true for the global inference relation. Assume now $G \vdash F$ is true and consider an arbitrary state s in an arbitrary Kripke structure $\mathcal{K} = (S, \models, \rho)$. If $s \not\models G$ then trivially $s \models G \to F$. If $s \models G$ then by $G \vdash F$ also $s \models F$. Again we get $s \models G \to F$.

Solution to Exercise 4.7.7

ad (1) The proof proceeds by induction on the complexity of π.
Let $\pi = a$ for $a \in A$ and $(s, s') \in \rho(a)$.
From $s \in S^A_{s_0}$ we derive the existence of some n with $s \in S^n_{s_0}$. By definition $s' \in S^{n+1}_{s_0}$. This finishes the initial case of the induction since $S^{n+1}_{s_0} \subseteq S^A_{s_0}$.
Let $\pi = \pi_1; \pi_2$ and $(s, s') \in \rho(\pi)$.
By the semantics definition there is $s'' \in S$ such that $(s, s'') \in \rho(\pi_1)$ and $(s'', s') \in \rho(\pi_2)$. By the induction hypothesis for π_1 we get $s'' \in S^A_{s_0}$ from $s \in S^A_{s_0}$. Starting from this the induction hypothesis for π_2 yields $s' \in S^A_{s_0}$.
Let $\pi = \pi_1 \cup \pi_2$ and $(s, s') \in \rho(\pi)$.
By the semantics definition either $(s, s') \in \rho(\pi_1)$ or $(s, s') \in \rho(\pi_2)$. Then the induction hypothesis for either π_1 or π_2 yields $s' \in S^A_{s_0}$.
Let $\pi = \pi_1^* \in \Pi$ and $(s, s') \in \rho(\pi)$.
By the semantics definition there is some n, $0 \leq n$ and there are $t_0, \ldots t_n$ with $t_0 = s$, $(t_i, t_{i+1}) \in \rho(\pi_1)$ for all $0 \leq i < n$ and $t_n = s'$. By repeated application of the induction hypothesis on π_1 we show $t_0 \in S^A_{s_0}, \ldots t_i \in S^A_{s_0}$, $\ldots t_n \in S^A_{s_0}$
Let $\pi = \mathit{con}?$ for some formula con and $(s, s') \in \rho(\pi)$.
By the semantics definition this implies $s = s'$ and nothing needs to be proved.

ad (2) The proof proceeds by induction on the complexity of F.
Let $F = p$ for a proposition variable p then

$$\begin{aligned}(\mathcal{K}, s) \models p &\Leftrightarrow (s, p) \in \vdash \\ &\Leftrightarrow (s.p) \in \vdash \cap (S^A_{s_0} \times \mathrm{PVar}) \\ &\Leftrightarrow (s.p) \in \vdash^A_{s_0} \\ &\Leftrightarrow (\mathcal{K}^A_{s_0}, s) \models p\end{aligned}$$

The induction step for the propositinal connectives follwos easily. So we consider next $F = [\pi]F_1$.

$$
\begin{array}{lcll}
(\mathcal{K}, s) \models F & \Leftrightarrow & (\mathcal{K}, s') \models F_1 \text{ for all } s' \text{ with } (s, s') \in \rho(\pi) & \text{semantics} \\
& \Leftrightarrow & (\mathcal{K}^A_{s_0}, s') \models F_1 \text{ for all } s' \text{ with } (s, s') \in \rho(\pi) & \text{Ind.Hyp and Part 1} \\
& \Leftrightarrow & (\mathcal{K}^A_{s_0}, s') \models F_1 \text{ for all } s' \text{ with } (s, s') \in \rho^A_{s_0}(\pi) & \text{Part 1} \\
& \Leftrightarrow & (\mathcal{K}^A_{s_0}, s') \models F & \text{semantics}
\end{array}
$$

The case $F = \langle\pi\rangle F_1$ follows analogously.

Solution to Exercise 4.7.8

We start with the easier implication. So assume $\vdash [(a_1 \cup \ldots \cup a_n)^*]F_1 \to F_2$ and for some PDL Kripke structure $\mathcal{K} = (S, \vdash, \rho)$ assume $(\mathcal{K}, s) \models F_1$ for all $s \in S$ with the aim to show $(\mathcal{K}, s) \models F_2$ for all $s \in S$. For any $s' \in S$ we know $(\mathcal{K}, s') \models [(a_1 \cup \ldots \cup a_n)^*]F_1 \to F_2$. Since $(\mathcal{K}, s) \models F_1$ is true for all $s \in S$ we obtain $(\mathcal{K}, s') \models [(a_1 \cup \ldots \cup a_n)^*]F_1$ and thus $(\mathcal{K}, s') \models F_2$.
For the reverse implication assume $F_1 \vdash_G F_2$ and for an arbitrary PDL Kripke structure $\mathcal{K} = (S, \vdash, \rho)$, $s_0 \in S$ assume $(\mathcal{K}, s_0) \models [(a_1 \cup \ldots \cup a_n)^*]F_1$ with the goal to show $(\mathcal{K}, s_0) \models F_2$.
We observe first that $\{s' \in S \mid (s_0, s') \in \rho((a_1 \cup \ldots \cup a_n)^*)\}$ equals $S^A_{s_0}$ as defined in Exercise 4.7.7. Thus we know $(\mathcal{K}, s') \models F_1$ for all $s' \in S^A_{s_0}$. Exercise 4.7.7 entails $(\mathcal{K}^A_{s_0}, s') \models F_1$ for all $s' \in S^A_{s_0}$. The assumption, $F_1 \vdash_G F_2$, now implies $(\mathcal{K}^A_{s_0}, s') \models F_2$ for all $s' \in S^A_{s_0}$. Another appeal to 4.7.7 gives $(\mathcal{K}, s_0) \models F_2$ as desired.

Solution to Exercise 4.7.9

Let $F \in PModFml$ be a modal propositional formula as defined in Definition 34 and $a \in AP$ an atomic program. To decide wether F is satisfiable we built a formula $F^* \in PFml$ such that $(F^*)^a = F$, where $(F^*)^a$ is as defined in Exercise 4.7.1. This exercise guarantees

$$F \text{ is satisfiable in modal logic } \mathbf{K} \quad \text{iff} \quad F^* \text{ is satisfiable in PDL}$$

Since we know that satisfiability of PDL is decidable we are then finished. The construction of the transformed formula $F^* \in PFml$ is simple, e.g. $(F_1 \wedge F_2)^* = F_1^* \wedge F_2^*$.
The only non-trivial cases are

$$
\begin{array}{lcl}
(\Box F)^* & = & [a]F^* \\
(\Diamond F)^* & = & \langle a\rangle P^*
\end{array}
$$

Solution to Exercise 4.7.10

Use the same set-up as in the solution to Exercise 4.7.9 but with the following translation:

$(\Box A)^* \quad = \quad [a^*]A^*$
$(\Diamond A)^* \quad = \quad \langle a^* \rangle A^*$

Again the desired property, equivalence of satisfiability, is not hard to see.

Solution to Exercise 4.7.11

This is easy. To decide $F_1 \vdash_G F_2$ let $n = max\{md(F_1, md(F_2)\}$ and find out whether $({}_1 \wedge \Box F_1 \ldots \Box^n F_1) \rightarrow F_2$ is satisfiable. If and only if that is the case $F_1 \vdash_G F_2$ is true. This is the statement proved in Exercise 3.9.14.

Solution to Exercise 4.7.12

This is again easy. To decide $F_1 \vdash_G F_2$ let $A = \{a_1, \ldots, a_n\}$ be all atomic programs occuring in F_1 or F_2. By Exercise 4.7.8 it sufices to find out whether $[(a_1 \cup \ldots \cup a_n)^*]F_1 \rightarrow F_2$ is universally valid in PDL. This can be done, as Theorem 103 assures us.

Solutions to Chapter 5

Solution to Exercise 5.6.1 The answer will be **no**. We will even show that the binary relation $y = 2 * x$ is not definable in the theory in question. The proof proceeds by contradiction. So we assume that $y = 2 * x$ is in fact definable.

Let V be the vocabulary with two letters a and b, $V = \{a, b\}$. By $L_{a2a} \subset V^\omega$ we denote the set of omega words defined by $FOW \wedge \exists x \exists y (y = 2 * x \wedge a(x) \wedge a(y) \wedge \forall z (z \neq x \wedge z \neq y \rightarrow b(y))$. By Theorem 114 there is a Büchi automaton $\mathcal{B}_{2a2}$ with $L^\omega(\mathcal{B}_{2a2}) = L_{a2a}$. By w_n we denote the word with

$$w_n(i) = \begin{cases} a & \text{if } i = n \text{ or } i = 2 * n \\ b & \text{otherwise} \end{cases}$$

Using this notation we may write $L_{a2a} = \{w_n \mid n \geq 0\}$. We may write

$$L_{a2a} = \bigcup_{i=0}^{i=k} K_i L_i^\omega \tag{6.1}$$

where $K_i, L_i \subseteq V^*$ are regular languages. See e.g. [Thomas, 1990, formula (1.1)] or [Schmitt, 2012, Satz 10.25].

There will be i_0, $0 \leq i_0 \leq k$ such that $w_n \in K_{i_0} L_{i_0}^\omega$ for infinitely many n, say $K_{i_0} L_{i_0}^\omega = \{w_n \mid n \in N\}$ for an infinite subset $N \subseteq \mathbb{N}$.

If a word in L_{i_0} contains the letter a then there would be words in $L_{i_0}^\omega$ containing infinitely many occurences of a. Since any word in L_{a2a} has exactly two occurences of a all words in L_{i_0} contain only the letter b. Thus K_{i_0} has the following two properties

1. For all $n \in N$ there is $w \in K_{i_0}$ of length $\geq 2 * n + 1$ with $w(n) = a$, $w(2 * n) = a$, and $w(m) = b$ otherwise
2. For all $w \in K_{i_0}$ there is $n \in N$ with $w(n) = a$, $w(2 * n) = a$, and $w(m) = b$ otherwise

We may now use the pumping lemma for regular languages to show that this is impossible. Thus arriving at the desired contradiction.

∎

Solution to Exercise 5.6.2 Let $\mathcal{T} = (\mathbb{N}, <, \xi)$ be an arbitrary omega structure.

1 If $\xi \models (F_1 \mathbf{U} G) \wedge (F_2 \mathbf{U} G)$ then there are n_1 and n_2 such that
$\xi_{n_1} \models G$ and $\xi_{n_2} \models G$
$\xi_m \models F_1$ for all m with $0 \leq m < n_1$
$\xi_m \models F_2$ for all m with $0 \leq m < n_2$
For $n = min(n_1, n_2)$ we get $\xi_n \models G$ and $\xi_m \models F_1 \wedge F_2$ for all m with $0 \leq m < n$. The reverse impication is obvious.

2 $\xi \models F \mathbf{U} (G_1 \vee G_2)$ iff

there is n such that $\xi_n \models (G_1 \vee G_2)$ and $\xi_m \models F$ for all m with $0 \leq m < n$.

This is trivially equivalent to

there is n such that $\xi_n \models G_1$ or $\xi_n \models G_2$ and $\xi_m \models F$ for all m with $0 \leq m < n$.

This is exactly the definition of $\xi \models (F \mathbf{U} G_1) \vee (F \mathbf{U} G_2)$

∎

Solution to Exercise 5.6.3

1. $\Box(p \rightarrow \Diamond q)$
2. $\Box(p \rightarrow (\neg q \wedge \mathbf{X} \neg q \wedge \mathbf{XX} \neg q \wedge \Diamond q))$

Solution to Exercise 5.6.4 Assume there is an LTL formula G that expresses the stated property. Let $\mathcal{A}_G$ be the Büchi automaton associated with G as guaranteed by Theorem 120. Obviously, there is at least one accepting computation sequence t satisfying $t \models G$. By Lemma 122 there is also a finite, cyclic accepting computation sequence s_e satisfying $s_e \models G$. Let $s_e = s_e^1, s_e^2, \ldots, s_e^n, \ldots$. Let n_i be all positions in increasing order such that $s_e^{n_i} \models p$. We know there are infinitely many. Let m_i be the least index $m_i \geq n_i$ such that $s_e^{m_i} \models q$. By the property encoded by G we should have $(m_{i+1} - n_{i+1}) > (m_i - n_i)$. This is not possible for the cyclic sequence s_e.

Solution to Exercise 5.6.5

safety $s_0 \models \mathbf{AG} \bigwedge_{1 \leq i < j \leq n} \neg(c_i \wedge c_j)$

liveness $s_0 \models \mathbf{AG} \bigwedge_{i\in\{1,\ldots,n\}} (t_i \rightarrow \mathbf{EF} c_i)$

non-blocking $s_0 \models \mathbf{AG} \bigwedge_{i\in\{1,\ldots,n\}} (\mathbf{EF} t_i)$

non-sequencing $\mathbf{EF} \bigwedge_{1\leq i\leq n} (t_i \wedge \bigvee_{1\leq j\leq n \& i\neq j} \mathbf{EXE}(\neg c_i \ \mathbf{U} \ c_j))$

non-alternating $\mathbf{EF} \bigwedge_{1\leq i\leq n} (c_i \wedge \mathbf{EXE}((\bigwedge_{1\leq j\leq n \& i\neq j} \neg c_j) \ \mathbf{U} \ c_i)))$

Solution to Exercise 5.6.6

For $1 \leq i \leq p$ let $K_{p,q,i}$ be the set of all words $w \in V_{p,q}^{\omega}$ such that there is exactly one position n such that the i-th position in the letter $w(n)$ equals 1. Obviously

$$K_{p,q} = \bigcap_{1\leq i\leq p} K_{p,q,i}$$

Let $B_i \subseteq V_{p,q}$ be the set of all letters whose i-th position is 0 and C_i its complement $C_i = V_{p,q} \setminus B_i$. Then $K_{p,q,i} = B_i^* C_i B_i^{\omega}$ which shows that $K_{p,q,i}$ is omega-regular. Since omega-regular sets are closed under intersection also $K_{p,q}$ is omega-regular. If you prefer a Büchi automaton over omega-regular expressions here it is:

Solution to Exercise 5.6.7

Let $\mathcal{B} = (S, V_1, s_0, \delta, F)$ be a Büchi automaton such that $L^{\omega}(\mathcal{B}) = L$. We will produce a Büchi automaton $\mathcal{B}_{\mu} = (S, V_2, s_0, \delta_{\mu}, F)$ such that $L^{\omega}(\mathcal{B}_{\mu}) = \mu(L)$. The only difference is in the definition of δ_{μ}:

$$\delta_{\mu}(s, c) = \bigcup \{\delta(s, d) \mid \mu(d) = c\}$$

$$\begin{array}{lll} w \in L^{\omega}(\mathcal{B}) & \text{iff} & \text{there is a sequence } (s_n)_{0\leq n} \text{ of states such that} \\ & & \text{for all } n: \quad s_{n+1} \in \delta(s_n, w(n)) \\ & \Rightarrow & \text{there is a sequence } (s_n)_{0\leq n} \text{ of states such that} \\ & & \text{for all } n: \quad s_{n+1} \in \delta_{\mu}(s_n, \mu(w(n))) \\ & \text{iff} & \mu(w) \in L^{\omega}(\mathcal{B}_{\mu}) \end{array}$$

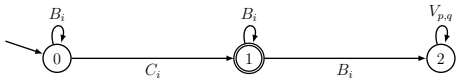


Figure 6.3: A Büchi automaton accepting $K_{p,q}$

$$
\begin{array}{lll}
w' \in L^\omega(\mathcal{B}_\mu) & \text{iff} & \text{there is a sequence } (s_n)_{0\leq n} \text{ of states such that} \\
 & & \text{for all } n: \quad s_{n+1} \in \delta_\mu(s_n, w'(n)) \\
 & \Rightarrow & \text{there is a sequence } (s_n)_{0\leq n} \text{ of states such that} \\
 & & \text{for all } n: \quad s_{n+1} \in \delta(s_n, w(n)) \\
 & & \text{for an appropriate } w \\
 & \text{iff} & w \in L^\omega(\mathcal{B})
\end{array}
$$

The appropriate w is obtained as follows. From $s_{n+1} \in \delta_\mu(s_n, w'(n))$ we get by definition of δ_μ some $d \in V_1$ with $\mu(d) = w'(n)$ and $s_{n+1} \in \delta(s_n, d)$. We set $w(n) = d$. By construction we see $\mu(w) = w'$.
In total we have shown $\mu(L^\omega(\mathcal{B})) = L^\omega(\mathcal{B}_\mu)$.

▪

Solution to Exercise 5.6.9

Let $L = \{S \subseteq G \mid f(S) \subseteq S \text{ and } U_0 \subseteq S\}$. As in the proof of Theorem 135 we can prove for $U = \bigcap L$ that $f(U) \subseteq U$. Thus by monotonicity of f we also obtain $f(f(U)) \subseteq f(U)$. Since all sets in L are supersets of U_0 we also have $U_0 \subseteq U$. By monotonicity we get $f(U_0) \subseteq f(U)$ and by the assumption of the theorem also $U_0 \subseteq f(U)$. This yields $f(U) \in L$ and therefore $U \subseteq f(U)$. Alltogether $U = f(U)$.
Let now W be a fixed point above U_0, i.e., $f(W) = W$ and $U_0 \subseteq W$. We get $W \in L$ by definition and thus $U \subseteq W$.

▪

Solution to Exercise 5.6.10

You need to require $f(U_0) \subseteq U_0$. The proof is the *dual* of the proof of Exercise 5.6.9.

▪

Solution to Exercise 5.6.11

By Lemma 132 $\mathbf{AGEF}p$ is only expresible in LTL if $\mathbf{AGEF}p \leftrightarrow \mathbf{AGF}p$ is true. It can be seen that for the transition system in Figure 6.4 we have $s_0 \models \mathbf{AGEF}p$ but $s_0 \not\models \mathbf{AGF}p$.

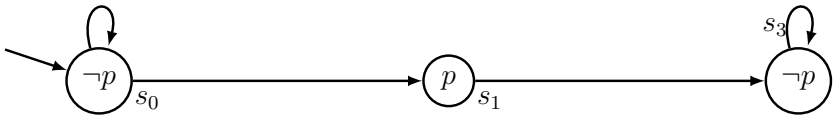


Figure 6.4: Transition system for **AGEF**p

■

Solution to Exercise 5.6.12

ad 1 We argue as follows

$$\begin{array}{rcl} X \subseteq Y & \Rightarrow & (D \setminus Y) \subseteq (D \setminus X) \\ & \Rightarrow & F(D \setminus Y) \subseteq F(D \setminus X) \\ & \Rightarrow & D \setminus F(D \setminus X) \subseteq D \setminus F(D \setminus Y) \\ & \Rightarrow & G(X) \subseteq G(Y) \end{array}$$

ad 2 We make us of the characteristion of the least fixed point of G obtained in the proof of Theorem 135 as the intersection of all $X \subseteq D$ with $G(X) \subseteq X$, i.e.,

$$lfp(G) = \bigcap \{X \mid X \subseteq D \text{ and } G(X) \subseteq X\}$$

By definition of G we have $G(X) \subseteq X$ if and only if $(D \setminus X) \subseteq F(D \setminus X)$. We may thus equivalently rewrite the definition of $lfp(G$ as

$$\begin{array}{rcl} lfp(G) & = & \bigcap \{D \setminus Y \mid Y \subseteq D \text{ and } Y \subseteq F(Y)\} \\ & = & D \setminus \bigcup \{D \setminus Y \mid Y \subseteq D \text{ and } Y \subseteq F(Y)\} \\ & = & D \setminus gfp(F) \end{array}$$

ad 3 analogous to part 2.

■

Solution to Exercise 5.6.13

Let $\mathcal{B} = (S, V_{p,q}, s_0, \delta, F)$ be the given Büchi automaton with edge vocabulary $V_{p,q}$ and

$$L^\omega(\mathcal{B}) = \{w \in K_{p,q} \mid w^0 \models \phi[s_{w,1}, \ldots, s_{w,p}, S_{w,1}, \ldots, S_{w,q}]\}$$

for the given monadic second-order formula ϕ.

The Büchi automaton $\mathcal{B}_{ex}$ to be constructed works over the alphabet $V_{p-1,q}$ so we need some notation to relate letters from the two different alphabets. For $a \in V_{p-1,q}$ we denote by $a \downarrow x$ the letter in $V_{p,q}$ with x inserted at the position $1+p$. Formally

$$(a \downarrow x)[i] = \begin{cases} a[i] & \text{if } 0 \leq i \leq p \\ x & \text{if } i = 1+p \\ a[i+1] & \text{if } p+1 < i \leq p+q+1 \end{cases}$$

Of course, only $x \in \{0,1\}$ make sense in the present context. Dually we done for $a \in V_{p,q}$ by $a \uparrow$ the letter in $V_{p-1,q}$ obtained from a by dropping its $p+1$ component. Formally

$$(a \uparrow)[i] = \begin{cases} a[i] & \text{if } 0 \leq i < p \\ a[i+1] & \text{if } p \leq i \leq p+q \end{cases}$$

We set

$$\begin{array}{lcl} S_{ex} & = & S \\ s_0^{ex} & = & s_0 \\ F_{ex} & = & F \\ \delta_{ex}(s,a) & = & \{s' \mid s' \in \delta(s, a \downarrow 0)\} \cup \{s' \mid s' \in \delta(s, a \downarrow 1)\} \end{array}$$

Consider now $w \in L^\omega(\mathcal{B}_{ex})$. By definition of acceptance there is a run $s_0, \ldots, s_n \ldots$ of $\mathcal{B}_{ex}$ such that a final state $q \in S$ occurs infinitely often among the s_n and for all n we have $s_{n+1} \in \delta_{ex}(s_n, w(n))$. By definition of δ_{ex} there are $x_n \in \{0,1\}$ such that $s_{n+1} \in \delta(s_n, w(n) \downarrow x_n)$. If we define $w' \in V_{p,q}$ by $w'(n) = w(n) \downarrow x_n$ we obtain $w' \in L^\omega(\mathbf{B})$. By assumption this entails

$$(w')^0 \models \phi[s_{w',1}, \ldots, s_{w',p}, S_{w',1}, \ldots, S_{w',q}]$$

and thus also

$$(w')^0 \models \exists x_p \phi[s_{w',1}, \ldots, x_{w',p-1}, S_{w',1}, \ldots, X_{w',q}]$$

Since $(w')^0 = w^0$, $s_{w',i} = s_{w,i}$ for $1 \leq i < p$ and $S_{w',j} = S_{w,j}$ for $1 \leq j \leq q$ we also have

$$w^0 \models \exists x_p \phi[s_{w,1}, \ldots, s_{w,p-1}, S_{w,1}, \ldots, S_{w,q}]$$

This already shows

$$L^\omega(\mathcal{B}_{ex}) \subseteq \{w \in K_{p-1,q} \mid w^0 \models \exists x_p \phi[s_{w,1}, \ldots, s_{w,p-1}, S_{w,1}, \ldots, S_{w,q}]\}$$

Now, consider $w \in V^{\omega}_{p-1.q}$ such that $w^0 \models \exists x_p \phi[s_{w,1}, \ldots, s_{w,p-1}, S_{w,1}, \ldots, S_{w,q}]$ Thus for some n $w^0 \models \phi[s_{w,1}, \ldots, s_{w,p-1}, n, S_{w,1}, \ldots, S_{w,q}]$. We can easily find $w^+ \in V^{\omega}_{p.q}$ with $(w^+(i)) \upharpoonright = w(i)$ for all $i \in \mathbb{N}$ and $s_{w^+,p} = n$. Thus $(w^+)^0 \models \phi[s_{w^+,1}, \ldots, s_{w^+,p-1}, s_{w^+,p}, S_{w^+,1}, \ldots, S_{w^+,q}]$. By assumption $w^+ \in L^{\omega}(\mathcal{B})$. Thus there is a run $s_0, \ldots, s_k, \ldots$ of $\mathcal{B}$ with $s_{k+1} \in \delta(s_k, w^*(k))$ for all k and a state $q \in F$ occuring infinitely often among the s_k. By definition of δ_{ex} we also have $s_{k+1} \in \delta_{ex}(s_k, w(k))$ for all k, i.e. $w \in L^{\omega}(\mathcal{B}_{ex})$.

∎

Bibliography

[Abate *et al.*, 2007] Pietro Abate, Rajeev Goré, & Florian Widmann. An on-the-fly tableau-based decision procedure for PDL-satisfiability. *CoRR, Computing Research Repository of Cornell University Library*, abs/0711.1016, 2007.

[Ackermann, 1935] Wilhelm Ackermann. Untersuchungen über das Eliminationsproblem der mathematischen Logik. *Mathematische Annalen*, 110:390–413, 1935.

[Allemang & Hendler, 2008] Dean Allemang & Jim Hendler. *Semantics Web for the Working Ontologist*. Morgan Kaufmann, 2008.

[Baader *et al.*, 2003] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, & Peter Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.

[Barnett *et al.*, 2005] Michael Barnett, Bor-Yuh Evan Chang, Robert DeLine, Bart Jacobs 0002, & K. Rustan M. Leino. Boogie: A modular reusable verifier for object-oriented programs. In Frank S. de Boer, Marcello M. Bonsangue, Susanne Graf, & Willem P. de Roever, editors, *FMCO*, volume 4111 of *Lecture Notes in Computer Science*, pages 364–387. Springer, 2005.

[Barwise, 1977] Jon Barwise, editor. *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co, 1977.

[Beckert *et al.*, 2007] Bernhard Beckert, Reiner Hähnle, & Peter H. Schmitt, editors. *Verification of Object-Oriented Software: The KeY Approach*. LNCS 4334. Springer-Verlag, 2007.

[Biere *et al.*, 1999] A. Biere, A. Cimatti, E. Clarke, & Y. Zhu. Symbolic model checking without bdds. In *Proc. of the Workshop on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'99)*, LNCS. Springer-Verlag, 1999.

[Biere *et al.*, 2003] A. Biere, A. Cimatti, E. Clarke, O. Strichman, & Y. Zhu. Bounded model checking. *Advances in Computers,*, 58, 2003.

[Bull & Segerberg, 1984] Robert A. Bull & Krister Segerberg. Basic modal logic. In *Handbook of Philosophical Logic*, volume II Extensions of Classical Logic, pages 1 – 88. D.Reidel, 1984.

[Börger *et al.*, 1982] Egon Börger, Erich Grädel, & Yuri Gurevich. *The Classical Decision Problem*. Perspectives in Mathematical Logic. "spv", 1982.

[Clarke & Draghicescu, 1988] Edmund M. Clarke & I. A. Draghicescu. Expressibility results for linear-time and branching-time logics. In J. W. de Bakker, Willem P. de Roever, & Grzegorz Rozenberg, editors, *REX Workshop*, volume 354 of *Lecture Notes in Computer Science*, pages 428–437. Springer, 1988.

[Clarke *et al.*, 1986] E.M. Clarke, E.A. Emerson, & A.P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, 1986.

[Clarke *et al.*, 1993] E. M. Clarke, O. Grumberg, & D. Long. Verification tools for finite state concurrent systems. In de Bakker et al. [de Bakker *et al.*, 1993], pages 124 – 175.

[Clarke *et al.*, 2001] Edmund M. Clarke, Rna Grumberg, & Doron A. Peled. *Model Checking*. The MIT Press, 2001.

[Conradie *et al.*, 2006a] Willem Conradie, Valentin Goranko, & Dimiter Vakarelov. Algorithmic correspondence and completeness in modal logic. i. the core algorithm sqema. *CoRR*, abs/cs/0602024, 2006.

[Conradie *et al.*, 2006b] Willem Conradie, Valentin Goranko, & Dimiter Vakarelov. Algorithmic correspondence and completeness in modal logic. ii. polyadic and hybrid extensions of the algorithm sqema. *J. Log. Comput.*, 16(5):579–612, 2006.

[Conradie *et al.*, 2009] Willem Conradie, Valentin Goranko, & Dimiter Vakarelov. Algorithmic correspondence and completeness in modal logic. iii. extensions of the algorithm sqema with substitutions. *Fundam. Inform.*, 92(4):307–343, 2009.

[Copty *et al.*, 2001] F. Copty, L. Fix, R. Fraer, E. Giunchiglia, G. Kamhi, A. Tacchella, & M.Y. Vardi. Benefits of bounded model checking at an industrial setting. In *Proc. 12th Intl. Conference on Computer Aided Verification (CAV'01)*, LNCS, pages 436–453. Springer, 2001.

[de Bakker *et al.*, 1993] J.W. de Bakker, W.P. de Roever, & G. Rozenberg, editors. *A Decade of Concurrency - Reflections and Perspectives*, volume 803 of *Lecture Notes in Computer Science*. Springer-Verlag, June 1993.

[Doherty *et al.*, 1997] P. Doherty, W. Łukasiewicz, & A. Szałas. Computing circumscirption revisited. *J.of automated reasoning*, 18(3):297–335, 1997.

[Emerson, 1992] E. Allen Emerson. Temporal and modal logic. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science. Volume B. Formal Models and Semantics*, pages 996–1072. Elevesier, 1992.

[Fitting, 1983] Melvin Fitting. *Proof Methods for Modal and Intuitionistic Logic*. D. Reidel, 1983.

[Fitting, 1990] Melvin C. Fitting. *First-Order Logic and Automated Theorem Proving*. Springer, 1990.

[Fitting, 1993] Melvin Fitting. Basic modal logic. In *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume Vol. 1 Logical Foundations, pages 368 – 448. Clarendon Press, 1993.

[Garson, 1984] James W. Garson. Quantification in modal logic. In *Handbook of Philosophical Logic*, volume II Extensions of Classical Logic, pages 249 – 308. D.Reidel, 1984.

[Glimm *et al.*, 2008] B. Glimm, I.Horrocks, C. Lutz, & U.Sattler. Conjunctive query answering for the description logic SHIQ. *J. of Artificial Intelligence Research*, 31:151–198, 2008.

[Goldblatt, 1982] Robert Goldblatt. *Axiomatising the logic of computer programming*, volume 130 of *LNCS*. Springer-Verlag, 1982.

[Goldblatt, 1987] Robert Goldblatt. *Logics of Time and Computation*. Number 7 in CSLI Lecture Notes. CSLI, 1987.

[Grädel, 1999] Erich Grädel. On the restraining power of guards. *J. Symb. Log.*, 64(4):1719–1742, 1999.

[Gustafsson, 1996] J. Gustafsson. An implementation and optimization of an algorithm for reducing formulas of second-order logic. Tech.Report LiTH-MAT-R-96-04, University of Linköping, Sweden, 1996.

[Halmos, 1974] Paul Richard Halmos. *Naive Set Theory*. Springer-Verlag Telos, 1974.

[Halmos, 1994] Paul Richard Halmos. *Naive Mengenlehre*. Vandenhoeck & Ruprecht, 1994.

[H.Andréka *et al.*, 1998] H.Andréka, J.van Benthem, & I.Németi. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27, 1998.

[Harel *et al.*, 2000] David Harel, Dexter Kozen, & Jerzy Tiuryn. *Dynamic Logic*. The MIT Press, 2000.

[Harel, 1979] David Harel. *First-Order Dynamic Logic*, volume 68 of *Lecture Notes in Computer Science*. Springer-Verlag, 1979.

[Harel, 1984] David Harel. Dynamic logic. In *Handbook of Philosophical Logic*, volume II Extensions of Classical Logic, pages 497 – 604. D.Reidel, 1984.

[Hayes, 2004] Patrick Hayes. Rdf semantics. Technical report, W3C, 2004.

[Hitzler *et al.*, 2009] Pascal Hitzler, Markus Krötzsch, & Sebastian Rudolph. *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC, 2009.

[Horrocks *et al.*, 2000] I. Horrocks, U. Sattler, & S. Tobies. Practical reasoning for very expressive description logics. *Logic Journal of the IGPL*, 8(3):239–263, 2000.

[Hughes & Cresswell, 1972] G. E. Hughes & M.J. Cresswell. *An Introduction to Modal Logic*. Methuen and Co Ltd, London, second edition, 1972.

[Hughes & Cresswell, 1984] G. E. Hughes & M.J. Cresswell. *A Companion to Modal Logic*. Methuen and Co Ltd, London, 1984.

[Huth & Ryan, 2000] Michael Huth & Mark Ryan. *Logic in Computer Science. Modelling and reasoning about systems*. Cambridge University Press, 2000.

[Kunen, 1977] Kenneth Kunen. *Combinatorics*, chapter B.3, pages 371–401. Volume 90 of Barwise [Barwise, 1977], 1977.

[Lamport, 1974] L. Lamport. A new solution of Dijkstra's concurrent programming. *Communications of the ACM*, 17(8):453–455, august 1974.

[Leino, 2008] K. Rustan M. Leino. This is Boogie 2. Working draft of the Boogie 2 language reference manual, 2008. http://research.microsoft.com/en-us/um/people/leino/papers.krml178.pdf.

[Lewis, 1918] C. I. Lewis. *A survey of symbolic logic*. University of California, Berkeley, 1918.

[Marx & Venema, 1997] Maarten Marx & Yde Venema. *Multi-Dimensional Modal Logic*, volume 4 of *Applied Logic Series*. Kluwer Academic Publishers, Dordrecht,Boston, London, 1997.

[M.Gabbay *et al.*, 2008] Dov. M.Gabbay, Renate A.Schmidt, & Andrzei Szałas. *Second-Order Quantifier Elimination. Foundations, Computational Aspects and Applications*, volume 12 of *Studies in Logic*. College Publication, 2008.

[Monk, 1976] J. Donald Monk. *Mathematical Logic*. Springer-Verlag, 1976.

[Mortimer, 1975] M. Mortimer. On languages with two variables. *Zeitschrift für mathematische Logik und Grundlagen*, 21:135–140, 1975.

[R.Drake, 1974] Frank R.Drake. *Set Theory. An Introduction to Large Cardinals*, volume 76 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, 1974.

[Rubin, 1967] Jean E. Rubin. *Set Theory for the Mathematician*. Holden-Day, 1967.

[Sahlqvist, 1975] H. Sahlqvist. Completeness and correspondance in first and second order semantics for modal logic. In S. Kanger, editor, *Proceedings of the Third Scandinavian Logic Colloquium*, pages 110–143. North Holland, 1975.

[Schild, 1991] Klaus Schild. A correspondence theory for terminological logics: preliminary report. In *Proc.of the 12.Int.Joint Conf. on Artifical Intelligence (IJCAI'91)*, pages 466–471, 1991.

[Schild, 1993] Klaus Schild. Combining terminological logics with tense logic. In *Proc. 6th Portuguese Conference on Artificial Intelligence, EPIA-93*, volume 727 of *Lecture Notes in Computer Science*, pages 105–120. Springer, 1993.

[Schmitt, 2008] Peter H. Schmitt. Formale Systeme. Lecture Notes, 2008. In German.

[Schmitt, 2012] Peter H. Schmitt. Formale Systeme. Lecture Notes, 2012. In German.

[Steinacker, 1990] Peter Steinacker. *Nichtklassische Logik*, chapter 3, pages 86 – 159. Akademie-Verlag, Berlin, 1990.

[Sterling, 1992] Colin Sterling. Modal and temporal logics. In Dov. M. Gabbay S. Abramsky & T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume Vol. 2 Background: Computational Structures, pages 478 – 563. Clarendon Press, 1992.

[Surányi, 1943] J. Surányi. Zur reduktion des entscheidungsproblems des logischen funktionenkalküls. *Mathematikai és Fizikai Lapok*, 50:51–74, 1943.

[Takeuti & Zaring, 1971] Gaisi Takeuti & Wilson M. Zaring. *Introduction to Axiomatic Set Theory*. Graduate Texts in Mathematics. Springer, 1971.

[Thomas, 1990] Wolfgang Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science*, volume B, Formal Models and Semantics, pages 135–191. Elsevier, 1990.

[van Benthem, 1984] Johan van Benthem. Correspondence theory. In *Handbook of Philosophical Logic*, volume II Extensions of Classical Logic, pages 167 – 248. D.Reidel, 1984.

[V.R.Pratt, 1976] V.R.Pratt. Semantical considerations on Floyd-Hoare logic. In *Proceedings of the 17 th IEEE Symp. Foundations of Computer Science*, pages 109 – 121. IEEE, 1976.