Tutorial
An introduction to metric semantics: operational and denotational models for programming and specification languages
Franck van Breugel
York University, Department of Computer Science, 4700 Keele Street, Toronto, Canada M3J 1P3
Received June 2000; revised September 2000; accepted September 2000
Communicated by M. Nivat

Abstract

Our focus is on the semantics of programming and specification languages. Over the years, different approaches to give semantics to these languages have been put forward. We restrict ourselves to the operational and the denotational approach, two main streams in the field of semantics. Two notions which play an important role in this paper are (non)determinism and (non)termination. Nondeterminism arises naturally in concurrent languages and it is a key concept in specification languages. Nontermination is usually caused by recursive constructs which are crucial in programming. The operational models are based on labelled transition systems. The definition of these systems is guided by the structure of the language. Metric spaces are an essential ingredient of our denotational models. We exploit the metric structure to model recursive constructs and to define operators on infinite entities. Furthermore, we also employ the metric structure to relate operational and denotational models for a given language. On the basis of four toy languages, we develop some general theory for defining operational and denotational semantic models and for relating them. This theory is applicable to a wide variety of languages. We start with a very simple deterministic and terminating imperative programming language. By adding the recursive while statement, we obtain a deterministic and nonterminating language. Next, we augment the language with the parallel composition resulting in a bounded nondeterministic and nonterminating language. Finally, we add some timed constructs. We obtain an unbounded nondeterministic and nonterminating specification language. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Metric semantics; Operational semantics; Denotational semantics; Programming language; Specification language; Nondeterminism; Nontermination; Labelled transition system; Metric space

E-mail address: franck@cs.yorku.ca (F. van Breugel).
1 Supported by Natural Sciences and Engineering Research Council of Canada.
1. Introduction

This paper presents an introduction to metric semantics for programming and specification languages. We concentrate on operational semantics, denotational semantics and proofs relating the two. Both in the definition of a denotational semantics and in the proof, metric spaces play a crucial role.

In an operational model, the semantics of a language is captured by means of an abstract machine. Here, we follow the predominant approach to operational semantics. We use labelled transition systems to model languages operationally. Labelled transition systems and operational semantics models are studied in Appendix B. For more details, we refer the reader to, for example, Hennessy’s textbook [31].

The main characteristic of a denotational semantics is its compositionality. The semantics of a composed statement is expressed in terms of the semantics of the statements of which it is composed. To deal with recursive constructs in the denotational setting, a mathematical structure with fixed points is needed. Traditionally, ordered spaces are exploited for that purpose. See, for example, Gunter’s textbook [28]. In this paper, we use metric spaces instead. The order-theoretic and metric approach have been contrasted by De Bakker and Rutten in [14, pp. 10,11].

Both in the definition of a denotational semantics and in the proof relating an operational and a denotational semantics Banach’s theorem plays a key role. This theorem ensures the existence of unique fixed points under certain conditions. In a denotational semantics, these fixed points are used to model recursive constructs. As we will see, an operational and a denotational semantics can be proved equal by showing that both are the unique fixed point of some function.

1.1. Related work

In the late seventies, metric spaces were introduced to the realm of semantics by Arnold and Nivat. They used them to model recursive program schemes denotationally. [43] is one of the first papers on metric spaces and semantics, which had significant impact on the area.

Since the early eighties, the Amsterdam Concurrency Group at CWI has been working on metric semantics. [14] contains a collection of papers of this group. De Bakker and De Vink wrote a textbook on metric semantics. [15] presents metric semantics for numerous language fragments and an extensive bibliography. In [16], the first paper of the group, De Bakker and Zucker showed how to solve recursive equations over metric spaces and exploited the solutions to model concurrent language features denotationally. [34] is another key paper of the group. In that paper, Kok and Rutten demonstrated how Banach’s fixed point theorem can be employed to define both operational and denotational semantic models and to relate these models. Metric semantics have been applied success-fully to real programming languages. See, for example, the work of America et al. [2, 3, 49] on metric semantics for the parallel object-oriented language POOL.

Members of the Programming Research Group at Oxford University have also been using metric spaces in semantics. In particular, they exploited them to model Timed
CSP denotationally. See, for example, the papers by Ouaknine and Reed [45] and Reed and Roscoe [47].

The group of Majster-Cederbaum at the University of Mannheim has worked on the comparison of order-theoretic and metric denotational semantic models. See, for example, the papers by Baier and Majster-Cederbaum [9] and Majster-Cederbaum and Zetzsche [40].

Metric spaces have been used to model probabilistic constructs. Baier and Kwiatkowska [8], Den Hartog [29], Norman [44] and De Vink and Rutten [52] exploited metric spaces to model recursive constructs in a probabilistic setting. Desharnais et al. [25] and Giacalone et al. [27] used metric spaces in a different way: the metric is employed to give a quantitative measure of the difference in semantics of probabilistic system specifications.

Metric spaces have also been used to deal with fairness. De Bakker and Zucker [17] modelled a language with fair merge along the lines of this paper. Rutten and Zucker [50] studied fairification of a metric space similar to the one we use in Section 4. Degano and Montanari [24] introduced a metric space such that the infinite computations which are Cauchy sequences in that space are the fair ones. Related results have been obtained by Costa [22] and Darondeau et al. [23]. Natarajan and Cleaveland [42] gave an alternative characterization of fair testing by means of a metric space.

In [26], Escardo presented a metric model of PCF. MacQueen et al. [39] gave a metric model for recursive polymorphic types for \( \lambda \)-calculus with constants.

1.2. Overview

In Appendices A–D, we present some general theory. This theory is applied to four toy languages in Sections 2–5. [20] contains solutions to the exercises which are distributed over the text. A way in which this paper could be traversed is given in the following diagram: start with the introduction of Section 2, followed by Section 2.1, then the introduction of Appendix B, etc.
2. Determinism and termination

For a very elementary language we develop an operational semantics and a denotational semantics. The language itself is not very interesting since it is too simple. However, this elementary language is well suited for illustrating various concepts which will also be exploited in later sections when we deal with more interesting – but also more complex – languages. The language we study here is deterministic, that is, every time we execute a given statement of the language it amounts to the same computation. The language is also terminating, that is, the execution of a statement always terminates.

The basic constructions of the language are the assignment statement and the skip statement. The assignment statement $v := e$ assigns, when executed, the value of the expression $e$ to the variable $v$. The skip statement skip is an inaction, that is, it does not change the values of any of the variables. From these basic constructions we can build more complex ones by means of the sequential composition and the if statement. The execution of the sequential composition $s_1 ; s_2$ starts with executing the statement $s_1$. Once the execution of $s_1$ has terminated, the statement $s_2$ starts. The execution of the if statement if $b$ then $s_1$ else $s_2$ fi amounts to performing the statement $s_1$ if the Boolean expression $b$ evaluates to true. If $b$ evaluates to false, then the execution continues with the statement $s_2$.

Based on the scheme presented in Appendix B, we define an operational semantics $\mathcal{O}$ for the language by means of a labelled transition system. The configurations of the system are statements accompanied by states. These states administrate the values of the variables. They are exploited when we evaluate the expressions and Boolean expressions. The states are also used to label the transitions of the system. A transition

$$[s, \xi] \xrightarrow{\xi'} [s', \xi']$$

models that the statement $s$ in the state $\xi$ can make a computation step resulting in the statement $s'$ and the state $\xi'$. These transitions are defined by means of a collection of axioms and rules. For example, the rule

$$
\frac{[s_1, \xi] \xrightarrow{\xi'} [s'_1, \xi']}{[s_1 ; s_2, \xi] \xrightarrow{\xi'} [s'_1 ; s_2, \xi']}
$$

expresses that if $[s_1, \xi]$ can make a transition to $[s'_1, \xi']$ labelled by $\xi'$ then $[s_1 ; s_2, \xi]$ can also make a transition labelled by $\xi'$ to $[s'_1 ; s_2, \xi']$. This structural approach to operational semantics is due to Plotkin [46]. For each construction, like the sequential composition, some axioms and rules are given, like the one above. The labelled transition system is shown to be deterministic and terminating. A system is deterministic if every configuration has at most one outgoing transition. If there are no infinite transition sequences of the form

$$[s_0, \xi_0] \xrightarrow{\xi_1} [s_1, \xi_1] \xrightarrow{\xi_2} \ldots$$
then the system is terminating. Based on this labelled transition system, the operational semantics assigns to each statement a function mapping every state to a finite sequence of states.

Besides the operational semantics we also give a denotational semantics $\mathcal{D}$ for the language. The key feature of a denotational semantics is its compositionality. The meaning of a composed statement is defined in terms of the meaning of the statements it is built from. For example, $\mathcal{D}(s_1; s_2)$ is given in terms of $\mathcal{D}(s_1)$ and $\mathcal{D}(s_2)$. Hence, its compositionality can be rephrased as follows. For each construct, like the sequential composition, there exists a corresponding semantic one. For the sequential composition we will introduce a semantic counterpart, also denoted by ‘;’, acting on functions from states to finite sequences of states such that

$$\mathcal{D}(s_1; s_2) = \mathcal{D}(s_1); \mathcal{D}(s_2).$$

Having introduced an operational semantics $\mathcal{O}$ and a denotational semantics $\mathcal{D}$ for the language, we are of course interested in their relationship. The two semantic models are shown to coincide. Having established this relation between the two semantic models, we can for example conclude that the operational semantics is also compositional.

The rest of this section is organised as follows. The language is presented in Section 2.1. In Section 2.2, the operational semantics is introduced. The denotational semantics is defined in Section 2.3. In Section 2.4, we relate the two semantic models.

2.1. Language definition

The basic constructions of the language we study in this section are the assignment statement and the skip statement. Statements are combined by means of the sequential composition and the if statement.

For the introduction of these constructs we presuppose
- a set \((v \in \text{Var})\) of variables,
- a set \((e \in \text{Exp})\) of expressions, and
- a set \((b \in \text{BExp})\) of Boolean expressions.

In the semantic models, we will treat these simply as abstract sets with no additional structure. In the examples, we will encounter expressions like $3 \times v$, $v + 1$, and $v \div 2$ and Boolean expressions like $v > 1$ and $\text{odd}(v)$.

**Definition 1.** The set \((s \in \text{Stat})\) of statements is defined by

$$s ::= v := e \mid \text{skip} \mid s ; s \mid \text{if } b \text{ then } s \text{ else } s \text{ fi}.$$ 

For example,

$$\text{if } \text{odd}(v) \text{ then } v := 3 \times v; v := v + 1 \text{ else } v := v \div 2 \text{ fi}$$

is a statement of the language.
2.2. Operational semantics

The operational semantics for the language is defined in terms of a labelled transition system. As we will see, this system is deterministic and terminating.

For a general introduction to labelled transition systems we refer the reader to Section B. A configuration of the labelled transition system at hand consists of a statement and a state. This statement is either as introduced in Definition 1 or it is the empty statement $E$. The latter we use to signal termination. We extend the set of statements with the empty statement in

**Definition 2.** The set $(\bar{s} \in) Stat_E$ is defined by

$$\bar{s} ::= s | E.$$ 

A state assigns to each variable its value. To keep things simple, we assume the values to be natural numbers. The states are used to store and retrieve the values of the variables.

**Definition 3.** The set $(\zeta \in) \Sigma$ of states is defined by

$$\Sigma = Var \rightarrow \mathbb{N}.$$ 

Assigning the natural number $n$ to the variable $v$ in the state $\zeta$ gives rise to the state $\zeta\{n/v\}$ defined by

$$\zeta\{n/v\}(w) = \begin{cases} n & \text{if } v = w, \\ \zeta(w) & \text{otherwise}. \end{cases}$$

In a configuration of the labelled transition system a statement is joined by a state. This state is needed to evaluate the expressions and Boolean expressions. To simplify the model, we make the following assumptions. The evaluation of an expression and a Boolean expression

- always terminates and delivers a value (that is, a natural number) and $true$ or $false$, respectively,
- delivers a unique result (that is, the evaluation is deterministic), and
- exhibits no side effects on the state (that is, the evaluation does not change the state).

These evaluations are modelled by the given semantic functions

$$\mathcal{E} : Exp \rightarrow \Sigma \rightarrow \mathbb{N}$$

and

$$\mathcal{B} : BExp \rightarrow \Sigma \rightarrow \{true, false\}.$$ 

With $\mathcal{E}(e)(\zeta)$ we denote the value of the expression $e$ in the state $\zeta$. The value of the Boolean expression $b$ in the state $\zeta$ is indicated by $\mathcal{B}(b)(\zeta)$. 
The actions of the labelled transition system are states. We have left to define the transition relation. According to Definition B.1, it is a subset of

\[(Stat_E \times \Sigma) \times \Sigma \times (Stat_E \times \Sigma)\]

It is specified by a collection of axioms and rules. An axiom is of the form

\[c \xrightarrow{a} c'\]

It specifies that for all configurations (of the form) \(c\) and \(c'\) and for all actions (of the form) \(a\), there is a transition from \(c\) to \(c'\) labelled by \(a\). The axioms are usually used to describe the basic constructions like the assignment statement and the skip statement. The configuration \(c\) in general contains these constructions. A rule of the form

\[c_1 \xrightarrow{a_1} \ldots \xrightarrow{a_n} c'_{n+1}\]


tells us that if \(c_i\) can make a transition to \(c'_i\) labelled by \(a_i\) for \(i = 1, \ldots, n\) then \(c\) can make a transition to \(c'\) labelled by \(a\). Most of the time, the rules are exploited to model the other constructions, like the sequential composition and the if statement. The configuration \(c\) is usually built from those constructions and the configurations \(c_1; \ldots; c_{n+1}\) (something like, for example, \(c_1; c_2\)). The transition relation is defined as the smallest subset satisfying the axioms and rules.

The transition relation is presented in

**Definition 4.** The transition relation \(\rightarrow\) is defined by the following axioms and rules:

1. \([v := e, \varsigma] \xrightarrow{[e, \varsigma\{n/v\}], \text{where } n = \delta(e)(\varsigma)}\)
2. \([\text{skip}, \varsigma] \xrightarrow{[e, \varsigma]}\]
3. \([s_1, \varsigma] \xrightarrow{[e, \varsigma', \varsigma''], \text{if } B(b)(\varsigma) = \text{true}}\)
4. \([s_1, \varsigma] \xrightarrow{[e, \varsigma', \varsigma''], \text{if } B(b)(\varsigma) = \text{false}}\]

**Exercise 5.** Prove that there exists a smallest subset of \((Stat_E \times \Sigma) \times \Sigma \times (Stat_E \times \Sigma)\) satisfying the axioms and rules of Definition 4.

Some remarks:
- A transition

\([s, \varsigma] \xrightarrow{\varsigma'}[\tilde{s}, \varsigma']\]
tells us that the statement \( s \) in the (current) state \( \varsigma \) can perform a computation step resulting in the statement \( \bar{s} \) and the (possibly updated) state \( \varsigma' \). Transitions of the form
\[
[s, \varsigma] \xrightarrow{\bar{s}} [\bar{s}, \varsigma''],
\]
with \( \varsigma' \neq \varsigma'' \), are not provable (see Exercise 7(1)). Consequently, the labels could be removed from the transitions. In that way, we would obtain a transition system rather than a labelled transition system. However, we keep the labels since it makes explicit which part of the configuration we observe in the operational semantics and because the labels will be useful in later sections.

- **Axiom (1)** expresses that the assignment \( v := e \) executed in the state \( \varsigma \) terminates after updating the state \( \varsigma \) by assigning to the variable \( v \) the value of the expression \( e \) in the state \( \varsigma \).
- The **skip statement** \( \text{skip} \) makes a \( \varsigma \)-transition (not changing the state) and terminates.
- **Rule (3)** should be interpreted as follows. If \( s_1 \) performs a computation step and terminates, then \( s_1 ; s_2 \) also makes the same step and then starts executing \( s_2 \). If \( s_1 \) performs a computation step and results in \( s_1' \), then \( s_1 ; s_2 \) also does that step and turns itself into \( s_1' ; s_2 \).
- The execution of the if statement \( \text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi} \) is governed by the value of the Boolean expression \( b \) in the current state \( \varsigma \). If \( b \) evaluates to \( \text{true} \) in \( \varsigma \) then \( s_1 \) is executed next. Otherwise, the execution continues with \( s_2 \).
- The terminal configurations are of the form \([E, \varsigma]\) (see Exercise 7(2)).

**Example 6.** Consider the statement
\[
s = \text{if } \text{odd}(v) \text{ then } v := 3 \times v; v := v + 1 \text{ else } v := v \div 2 \text{ fi}.
\]
If, for example, \( \varsigma(v) = 5 \), then we can prove the transition
\[
[s, \varsigma] \xrightarrow{} [v := v + 1, \varsigma\{15/v\}]
\]
as follows.
\[
[v := 3 \times v, \varsigma] \xrightarrow{} [E, \{15/v\}]
\]
\[
[v := 3 \times v; v := v + 1, \varsigma] \xrightarrow{} [v := v + 1, \varsigma\{15/v\}]
\]
\[
[s, \varsigma] \xrightarrow{} [v := v + 1, \varsigma\{15/v\}]
\]
Furthermore, we have that
\[
[v := v + 1, \varsigma\{15/v\}] \xrightarrow{} [E, \varsigma\{16/v\}].
\]

**Exercise 7.** Prove that
(1) if \([\bar{s}, \varsigma] \xrightarrow{} [\bar{s}', \varsigma'']\) then \( \varsigma' = \varsigma'' \) and
(2) \([\bar{s}, \varsigma] \rightarrow \) if and only if \( \bar{s} = E \).
Every configuration of the labelled transition system has at most one outgoing transition as is shown in

**Proposition 8.** The labelled transition system is deterministic.

**Proof.** We have to prove that for all \( s \in \text{Stat}_E \) and \( \zeta \in \Sigma \), the set

\[
\mathcal{L}([\bar{s}, \zeta]) = \{ \langle \zeta', [\bar{s}', \zeta'] \rangle \mid [\bar{s}, \zeta] \xrightarrow{\zeta'} [\bar{s}', \zeta'] \}
\]

contains at most one element. We show this by structural induction on \( \bar{s} \). Only two cases are elaborated on.

- Let \( \bar{s} = \text{skip} \). Obviously, the set \( \{ \langle \zeta, [E, \zeta] \rangle \} \) contains one element.
- Let \( \bar{s} = s_1 ; s_2 \). In this case we have that

\[
\mathcal{L}([s_1; s_2, \zeta]) = \{ \langle \zeta', [s_2, \zeta'] \rangle \mid \langle \zeta', [s_1, \zeta'] \rangle \in \mathcal{L}([s_1, \zeta]) \} 
\]

By induction, the set \( \mathcal{L}([s_1, \zeta]) \) contains at most one element. Consequently, the above set also has at most one element. \( \square \)

There are no infinite transition sequences of the form

\[
[s_0, \zeta_0] \xrightarrow{s_1} [s_1, \zeta_1] \xrightarrow{s_2} \cdots
\]

as we prove in the following proposition.

**Proposition 9.** The labelled transition system is terminating.

**Proof.** We define the complexity function \( \text{comp} : \text{Stat}_E \rightarrow \mathbb{N} \) by

\[
\begin{align*}
\text{comp}(E) &= 0 \\
\text{comp}(v := e) &= 1 \\
\text{comp}(\text{skip}) &= 1 \\
\text{comp}(s_1 ; s_2) &= \text{comp}(s_1) + \text{comp}(s_2) \\
\text{comp}(\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi}) &= \max\{\text{comp}(s_1), \text{comp}(s_2)\} + 1.
\end{align*}
\]

The natural number \( \text{comp}(\bar{s}) \) gives us an upper bound of the length of the (maximal) transition sequence started in \([\bar{s}, \zeta] \). We leave it to the reader to verify that

\[
\text{if } [\bar{s}, \zeta] \xrightarrow{\zeta'} [\bar{s}', \zeta'] \text{ then } \text{comp}(\bar{s}) > \text{comp}(\bar{s}')
\]  

(Exercise 10). Since the set of natural numbers is well-founded we can conclude that the system is terminating. \( \square \)

**Exercise 10.** Prove Eq. (1).
According to Definition B.12, the above introduced deterministic and terminating labelled transition system induces an operational semantics

\[ \mathcal{O} : \text{Stat}_E \times \Sigma \rightarrow \Sigma^*. \]

The definition of \( \mathcal{O} \) amounts to

\[ \mathcal{O}([\bar{s}, \zeta]) = \zeta_1 \zeta_2 \cdots \zeta_n \quad \text{if} \quad [\bar{s}, \zeta] = [\bar{s}_0, \zeta_0] \xrightarrow{\zeta_0}[\bar{s}_1, \zeta_1] \xrightarrow{\zeta_1} \cdots \xrightarrow{\zeta_{n-1}}[\bar{s}_n, \zeta_n]. \]

**Example 11.** According to Example 6, we can conclude that

\[ \mathcal{O}([s; &]) = \{15; v\} \zeta\{16; v\} \]

if \( \zeta(v) = 5 \).

In the next proposition we present an alternative characterisation of the operational semantics \( \mathcal{O} \) which will be exploited when we relate the operational semantics to a denotational one (see Theorem 19).

**Proposition 12.** For all \( \bar{s} \in \text{Stat}_E \) and \( \zeta \in \Sigma \),

\[ \mathcal{O}([\bar{s}, \zeta]) = \begin{cases} \varepsilon & \text{if} \quad [\bar{s}, \zeta] \xrightarrow{\varepsilon} \\ \zeta' \mathcal{O}([\bar{s}', \zeta']) & \text{if} \quad [\bar{s}, \zeta] \xrightarrow{\zeta'}[\bar{s}', \zeta']. \end{cases} \]

**Proof.** Trivial. \( \Box \)

We conclude with the definition of the operational semantics for statements.

**Definition 13.** The function \( \mathcal{O} : \text{Stat} \rightarrow \Sigma \rightarrow \Sigma^* \) is defined by

\[ \mathcal{O}(s) = \lambda \zeta. \mathcal{O}([s, \zeta]). \]

Given a statement \( s \) and an initial state \( \zeta \), the operational semantics \( \mathcal{O} \) gives us a finite sequence of states. This sequence records the subsequent states arising during the execution of the statement \( s \) started in the state \( \zeta \).

**Exercise 14.** Instead of administrating all the states that arise during the execution of a statement, one could also choose to record only the final state. Define this alternative operational semantics \( \mathcal{A} : \text{Stat} \rightarrow \Sigma \rightarrow \Sigma \) and establish its relationship with the above-introduced operational semantics \( \mathcal{O} \).

### 2.3. Denotational semantics

Like the operational semantics, the denotational semantics assigns to each statement a function mapping every state to a finite sequence of states. Rather than using a labelled transition system, we introduce for each syntactic construct of the language a corresponding semantic one.
To the basic constructions \( v := e \) and \( \text{skip} \) we associate the constants

\[
\lambda_{\xi, \varepsilon}\{n/v\}, \quad \text{where } n = \delta(e)(\varepsilon)
\]

and

\[
\lambda_{\xi, \varepsilon},
\]

respectively. Note that the operational semantics also assigns the above constants to the basic constructions \( v := e \) and \( \text{skip} \). The semantic correspondent of the if statement is defined straightforwardly: for \( \beta \in \Sigma \to \{\text{true}, \text{false}\} \) and \( f, g \in \Sigma \to \Sigma^* \) we take

\[
\lambda_{\xi, \varepsilon} \begin{cases} 
  f(\xi) & \text{if } \beta(\xi) = \text{true}, \\
  g(\xi) & \text{if } \beta(\xi) = \text{false}.
\end{cases}
\]

We have left to introduce a semantic sequential composition

\[
; : (\Sigma \to \Sigma^*) \times (\Sigma \to \Sigma^*) \to (\Sigma \to \Sigma^*).
\]

Intuitively, for \( f, g \in \Sigma \to \Sigma^* \) and \( \varepsilon \in \Sigma \), if the sequence \( f(\varepsilon) \) is nonempty, then the sequence \( (f; g)(\varepsilon) \) consists of the concatenation of the sequences \( f(\varepsilon) \) and \( g(\text{last}(f(\varepsilon))) \), where \( \text{last}(f(\varepsilon)) \) is the last element of \( f(\varepsilon) \). If the sequence \( f(\varepsilon) \) is empty, then \( (f; g)(\varepsilon) = g(\varepsilon) \).

To support inductive arguments (see proof of Theorem 19), the semantic sequential composition of \( f, g \in \Sigma \to \Sigma^* \) is defined by means of a function

\[
; : \Sigma \times \Sigma^* \times (\Sigma \to \Sigma^*) \to \Sigma^*
\]

as

\[
\lambda_{\varepsilon}(f(\varepsilon); g),
\]

where we write \( \sigma; \varepsilon\; g \) instead of \( ;(\varepsilon, \sigma, g) \). If the sequence \( \sigma \) is nonempty, then the sequence \( \sigma; \varepsilon\; g \) consists of the concatenation of the sequences \( \sigma \) and \( g(\text{last}(\sigma)) \). Otherwise, \( \sigma; \varepsilon\; g = g(\varepsilon) \). We use the state component \( \varepsilon \) to keep track of the last state of the sequence \( \sigma \) we have inspected so far.

**Definition 15.** The function \( ; : \Sigma \times \Sigma^* \times (\Sigma \to \Sigma^*) \to \Sigma^* \) is defined by

\[
\sigma; \varepsilon\; g = \begin{cases} 
  g(\varepsilon) & \text{if } \sigma = \varepsilon, \\
  \varepsilon'(\sigma'\varepsilon'\; g) & \text{if } \sigma = \varepsilon'\sigma'.
\end{cases}
\]

After having introduced the semantic constructions we are ready to give the denotational semantics.

**Definition 16.** The function \( \mathcal{D} : \text{Stat} \to \Sigma \to \Sigma^* \) is defined by

\[
\mathcal{D}(v := e) = \lambda_{\xi, \varepsilon}\{n/v\}, \quad \text{where } n = \delta(e)(\varepsilon),
\]

\[
\mathcal{D}(\text{skip}) = \lambda_{\xi, \varepsilon},
\]
\[ D(s_1; s_2) = \lambda \varsigma. D(s_1)(\varsigma) ; D(s_2), \]
\[ D(\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi}) = \lambda \varsigma. \begin{cases} D(s_1)(\varsigma) & \text{if } B(b)(\varsigma) = \text{true}, \\ D(s_2)(\varsigma) & \text{if } B(b)(\varsigma) = \text{false}. \end{cases} \]

We conclude this section with an example.

**Example 17.** If \( \varsigma(v) = 5 \), then
\[
D(\text{if odd } (v) \text{ then } v := 3 \times v; v := v + 1 \text{ else } v := v \text{ div } 2 \text{ fi})(\varsigma)
\]
\[
= D(v := 3 \times v; v := v + 1)(\varsigma)
\]
\[
= D(v := 3 \times v)(\varsigma) ; D(v := v + 1)
\]
\[
= \varsigma\{15/v\} ; D(v := v + 1)
\]
\[
= \varsigma\{15/v\} D(v := v + 1)(\varsigma\{15/v\})
\]
\[
= \varsigma\{15/v\} \varsigma\{16/v\}.
\]

**2.4. Relating \( \mathcal{O} \) and \( D \)**

We relate the operational semantics \( \mathcal{O} \) introduced in Section 2.2 and the denotational semantics \( D \) presented in Section 2.3 by proving that the two semantic models coincide. For that purpose, we extend the denotational semantics as follows.

**Definition 18.** The function \( D : Stat_\Sigma \times \Sigma \rightarrow \Sigma^* \) is defined by
\[
D([E, \varsigma]) = \varepsilon,
\]
\[
D([s, \varsigma]) = D(s)(\varsigma).
\]

This extended denotational semantics and the operational semantics (of configurations) are shown to be equal in

**Theorem 19.** For all \( \bar{s} \in Stat_\Sigma \) and \( \varsigma \in \Sigma \),
\[
\mathcal{O}([\bar{s}, \varsigma]) = D([\bar{s}, \varsigma]).
\]

**Proof.** We prove this theorem by induction on \( \text{comp}(\bar{s}) \). We only treat a few cases. The other cases are left to the reader as an exercise (see Exercise 20).
- If \( \bar{s} = v := e \) and \( \varepsilon(e)(\varsigma) = n \) then
\[
\mathcal{O}([v := e, \varsigma])
\]
\[
= \varsigma\{n/v\}
\]
\[
= D([v := e, \varsigma]).
\]
Let $\bar{s} = s_1 ; s_2$ and assume that $[s_1, \xi] \xrightarrow{\xi'} [E, \xi']$. In this case,

\[
\begin{align*}
\mathcal{C}([s_1 ; s_2, \xi]) &= \xi' \mathcal{C}([s_2, \xi']) \quad \text{[Proposition 12]} \\
&= \xi' \mathcal{D}([s_2, \xi']) \quad \text{[induction]} \\
&= \xi' ; \xi \mathcal{D}(s_2) \\
&= \mathcal{C}([s_1, \xi]) ; \xi \mathcal{D}(s_2) \\
&= \mathcal{D}([s_1; s_2, \xi]) \quad \text{[induction]} \\
&= \mathcal{D}([s_1 ; s_2, \xi]).
\end{align*}
\]

Let $\bar{s} = \text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi}$ and suppose that $B(b)(\bar{\xi}) = \text{true}$. Then

\[
\begin{align*}
\mathcal{C}([\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi}, \bar{\xi}]) &= \mathcal{C}([s_1, \xi]) \\
&= \mathcal{D}([s_1, \xi]) \quad \text{[induction]} \\
&= \mathcal{D}([\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi}, \xi]). \quad \square
\end{align*}
\]

Exercise 20. Prove the case that $\bar{s} = s_1 ; s_2$ and that $[s_1, \xi] \xrightarrow{\xi'} [s_1', \xi']$.

From the above theorem we can deduce that also the denotational semantics and operational semantics (of statements) coincide.

Corollary 21. $\mathcal{C} = \mathcal{D}$.

Proof. For all $s \in \text{Stat}$ and $\bar{\xi} \in \Sigma$,

\[
\begin{align*}
\mathcal{C}(s)(\bar{\xi}) &= \mathcal{C}([s, \bar{\xi}]) \\
&= \mathcal{D}([s, \bar{\xi}]) \quad \text{[Theorem 19]} \\
&= \mathcal{D}(s)(\bar{\xi}). \quad \square
\end{align*}
\]

3. Determinism and nontermination

The while statement is added to the language we studied in the previous section. Also for this extended language we present an operational and a denotational semantics. The while statement adds a considerable amount of expressiveness to the language. We now also encounter nonterminating computations. The extended language is still deterministic. As we will see, the operational semantics presented in the previous section can be extended straightforwardly to also deal with the while statement. To model the
extended language denotationally we need some new ingredients as we will see below. Also when we relate the two models we will exploit some new techniques.

The execution of the while statement \( \text{while } b \text{ do } s \text{ od} \) amounts to repeatedly performing the statement \( s \). If the Boolean expression \( b \) becomes false after zero or more executions of the statement \( s \), then the execution of the while statement terminates at that point. Otherwise, the statement gives rise to a nonterminating computation. For example, the execution of the statement \( \text{while true do skip od} \) does not terminate.

Like in the previous section, the labelled transition system defining the operational semantics is specified by a collection of axioms and rules. The while statement is handled by adding one axiom and two rules. The obtained system is shown to be deterministic and nonterminating. To capture also nonterminating computations, the operational semantics assigns to each statement a function mapping every state to a finite or (countably) infinite sequence of states.

Extending the denotational semantics is not as easy as enhancing the operational one. The main difficulty is to model the while statement compositionally. Let us look at an example. Suppose that \( s \) is a terminating statement. Consider now the while statement \( \text{while true do } s \text{ od} \). The execution of this statement amounts to repeatedly performing \( s \). Any semantic model, including an operational and a denotational one, should therefore equate the statements \( \text{while true do } s \text{ od} \) and \( s ; \text{while true do } s \text{ od} \).

Because the denotational semantics \( \mathcal{D} \) should also be compositional, we have that

\[
\mathcal{D}(\text{while true do } s \text{ od}) = \mathcal{D}(s ; \text{while true do } s \text{ od}) = \mathcal{D}(s) ; \mathcal{D}(\text{while true do } s \text{ od}).
\]

This tells us that the denotation of the while statement should be a fixed point of the function \( \lambda f. \mathcal{D}(s) ; f \). By using a mathematical structure with fixed points this might be accomplished. Here, we exploit metric spaces for that purpose. A metric space is a set together with a metric assigning to each pair of elements of the set their distance. In Appendix A, some theory on metric spaces is developed. The existence of fixed points will be based on Banach’s fixed point theorem. This theorem roughly tells us that a contractive function from a complete metric space to itself has a unique fixed point. Sequences carry a natural Baire metric given in terms of the length of their longest common prefix. Based on this metric, we can also supply the functions from states to state sequences with a metric structure. As we will see, the function \( \lambda f. \mathcal{D}(s) ; f \) is a contractive mapping from this complete metric space to itself. According to Banach’s theorem, this function has a unique fixed point denoted by \( \text{fix}(\lambda f. \mathcal{D}(s) ; f) \). Hence,

\[
\mathcal{D}(\text{while true do } s \text{ od}) = \text{fix}(\lambda f. \mathcal{D}(s) ; f)
\]

is the compositional description of the while statement we were looking for. Similarly, we will deal with the general case where the Boolean expression might differ from true.
In the proof that the operational and denotational semantics coincide we also exploit
the metric structure. Their coincidence cannot be proved by structural induction because
of the presence of the while statement as we will see. In the proof we use the unique
fixed point proof principle. This proof principle is based on Banach’s fixed point
theorem. In Appendix B, we develop some theory to prove operational semantic models
equal to other semantic models by uniqueness of fixed point. Here, we will employ
those results.

The present section has the same structure as the previous one. The language is
presented in Section 3.1. In Section 3.2 and 3.3 we develop the operational and deno-
tational semantics. The proof that the two coincide is given in Section 3.4.

3.1. Language definition

We extend the language defined in Definition 1 with the while statement.

**Definition 22.** The set \( s \in \text{Stat} \) of statements is defined by

\[
s ::= v := e \mid \text{skip} \mid s ; s \mid \text{if } b \text{ then } s \text{ else } s \text{ fi} \mid \text{while } b \text{ do } s \text{ od}
\]

The statement

\[
\text{while } v > 1 \text{ do if odd } (v) \text{ then } v := 3 \times v \text{; } v := v + 1 \text{ else } v := v \text{ div } 2 \text{ fi od}
\]
is an example of a statement of the extended language.

3.2. Operational semantics

Also the operational semantics of the extended language is defined by means of a
labelled transition system. The system is also deterministic but nonterminating as we
will show below.

The configurations and the actions of the labelled transition system are defined as
in Section 2.2. The transition relation is extended by adding one axiom and two rules
for the while statement.

**Definition 23.** The transition relation \( \rightarrow \) is defined by the following axioms and rules:

1. \( [v := e, \xi] \xrightarrow{(n/e)} [E, \xi\{n/v\}] \), where \( n = E(e)(\xi) \)
2. \( [\text{skip}, \xi] \xrightarrow{\xi} [E, \xi] \)
3. \( \frac{[s_1, \xi] \xrightarrow{\xi'} [E, \xi'']} {[s_1; s_2, \xi] \xrightarrow{\xi'} [s_2, \xi'']} \)
4. \( \frac{[s_1, \xi] \xrightarrow{\xi'} [s_1', \xi'']} {[\text{if } b \text{ then } s_1 \text{ else } s_2, \xi] \xrightarrow{\xi'} [s_1', \xi'']} \) if \( \Phi(b)(\xi) = \text{true} \)
The axioms (1) and (2) and the rules (3) and (4) have already been discussed in Section 2.2. Recall that the execution of the statement \( \text{while } b \text{ do } s \text{ od} \) gives rise to the repeated execution of the statement \( s \). This is expressed by the two rules of (5). If the Boolean expression becomes \( \text{false} \) then the execution terminates. The latter is modelled by the axiom of (5).

**Exercise 24.** Consider the rule

\[
\begin{align*}
\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi, } \xi, \zeta & \xrightarrow{\xi'} [s_2, \xi''] \\
[\text{while } b \text{ do } s \text{ od} \xi, \zeta] & \xrightarrow{\xi'} [s', \xi'']
\end{align*}
\]

Show that one can prove with (1)–(4), (6) exactly the same transitions as with (1)–(5). Note that the rule (6) is less structural than the axiom and rules given in (5).

Like in Section 2.2, every configuration of the labelled transition system has at most one outgoing transition.

**Proposition 25.** The labelled transition system is deterministic.

**Proof.** Similar to the proof of Proposition 8. \( \square \)

The recursive while statement introduces nontermination as is shown in the following example.

**Example 26.** The proof

\[
\begin{align*}
[\text{skip}, \xi, \zeta] & \xrightarrow{\xi'} [E, \xi] \\
[\text{while true do skip od}, \xi, \zeta] & \xrightarrow{\xi'} [\text{while true do skip od}, \xi, \zeta]
\end{align*}
\]

shows that the labelled transition system at hand is nonterminating.

According to Definition B.14, the above-introduced deterministic and nonterminating labelled transition system induces an operational semantics

\[
\emptyset : \text{State} \times \Sigma \rightarrow \Sigma^\infty.
\]
The definition of $\mathcal{O}$ amounts to

$$\mathcal{O}([\mathcal{S}, \mathcal{Z}]) = \begin{cases} \mathcal{S}_1 \mathcal{S}_2 \cdots \mathcal{S}_n & \text{if } [\mathcal{S}, \mathcal{Z}] = [\mathcal{S}_0, \mathcal{Z}_0] \overset{\mathcal{S}_1}{\rightarrow} [\mathcal{S}_1, \mathcal{Z}_1] \overset{\mathcal{S}_2}{\rightarrow} \cdots \overset{\mathcal{S}_n}{\rightarrow} [E, \mathcal{Z}_n] \\ \mathcal{S}_1 \mathcal{S}_2 \cdots & \text{if } [\mathcal{S}, \mathcal{Z}] = [\mathcal{S}_0, \mathcal{Z}_0] \overset{\mathcal{S}_1}{\rightarrow} [\mathcal{S}_1, \mathcal{Z}_1] \overset{\mathcal{S}_2}{\rightarrow} \cdots \end{cases}$$

**Example 27.** According to Example 26, we have that

$$\mathcal{O}([\text{while true do skip od}, \mathcal{Z}]) = \mathcal{Z}^\omega.$$ 

The definition of the operational semantics of statements is as before.

**Definition 28.** The function $\mathcal{O} : \text{Stat} \rightarrow \Sigma \rightarrow \Sigma^\infty$ is defined by

$$\mathcal{O}(s) = \mathcal{Z} \mathcal{O}([s, \mathcal{Z}]).$$

### 3.3. Denotational semantics

The denotational semantics assigns to each statement a function mapping each state to a finite or infinite sequence of states, just like the operational semantics. Like in Section 2.3, we introduce for each syntactic construct of the language a corresponding semantic one. The assignment statement and the skip statement can be dealt with as before. Also the semantic correspondent of the if statement is the obvious adaptation of the one we have seen before (see Definition 36). We have left to introduce a semantic sequential composition and the semantic counterpart of the while statement.

Intuitively, we want the semantic sequential composition

$$; : \Sigma \times \Sigma^\infty \times (\Sigma \rightarrow \Sigma^\infty) \rightarrow \Sigma^\infty$$

to satisfy

$$\sigma ; f = \begin{cases} f(\mathcal{Z}) & \text{if } \sigma = e, \\ \mathcal{Z}'(\sigma' ; f) & \text{if } \sigma = \mathcal{Z}' \sigma', \end{cases}$$

(see Definition 15). However, the above cannot be justified by induction on the length of the sequence $\sigma$, since we also consider infinite sequences. Instead, we will exploit Banach’s theorem (Theorem A.18) for the justification of the above. This is done as follows. First, we endow the set

$$\Sigma \times \Sigma^\infty \times (\Sigma \rightarrow \Sigma^\infty) \rightarrow \Sigma^\infty$$

with a complete metric. Next, we introduce a function from this complete metric space to itself. Finally, this function is shown to be contractive. According to Banach’s theorem, this contractive function from a complete metric space to itself has a unique fixed point: the intended semantic sequential composition.

We endow the set $\Sigma^\infty$ with the Baire metric (Example A.2(3)) and the set $\Sigma$ with the discrete metric (Example A.2(1)) – from now on we use the convention that if we have not specified a metric for a given set, then this set is assumed to be endowed
with the discrete metric. By means of the constructions described in Definition A.5 we obtain a metric for the set (2). Since the Baire metric and the discrete metric give rise to complete spaces (Proposition A.13) and the operations $\times$ and $\rightarrow$ preserve completeness (Proposition A.14), we have turned the set (2) into a complete metric space.

Next, we introduce a function from this complete metric space to itself. The set of functions from this space to itself is denoted by

\[ [\Sigma \times \Sigma^\infty \times (\Sigma \rightarrow \Sigma^\infty) \rightarrow \Sigma^\infty]. \]

**Definition 29.** The function $\Phi : [\Sigma \times \Sigma^\infty \times (\Sigma \rightarrow \Sigma^\infty) \rightarrow \Sigma^\infty]$ is defined by

\[
\Phi(\phi)(\zeta, \sigma, f) = \begin{cases} 
  f(\zeta) & \text{if } \sigma = e, \\
  \zeta' \phi(\zeta', \sigma', f) & \text{if } \sigma = \zeta' \sigma'.
\end{cases}
\]

Before we can apply Banach’s theorem, we have to verify that the above introduced function is a contraction.

**Proposition 30.** The function $\Phi$ is contractive.

**Proof.** We show that for all $\phi_1, \phi_2 \in \Sigma \times \Sigma^\infty \times (\Sigma \rightarrow \Sigma^\infty) \rightarrow \Sigma^\infty$,

\[ d(\Phi(\phi_1), \Phi(\phi_2)) \leq \frac{1}{2} \cdot d(\phi_1, \phi_2). \]

It suffices to prove that for all $\zeta \in \Sigma$, $\sigma \in \Sigma^\infty$, and $f \in \Sigma \rightarrow \Sigma^\infty$,

\[ d(\Phi(\phi_1)(\zeta, \sigma, f), \Phi(\phi_2)(\zeta, \sigma, f)) \leq \frac{1}{2} \cdot d(\phi_1, \phi_2). \]

We distinguish the following two cases.

- **If** $\sigma = e$ **then**

  \[
  d(\Phi(\phi_1)(\zeta, e, f), \Phi(\phi_2)(\zeta, e, f)) = d(f(\zeta), f(\zeta)) = 0 \leq \frac{1}{2} \cdot d(\phi_1, \phi_2).
  \]

- **If** $\sigma = \zeta' \sigma'$ **then**

  \[
  d(\Phi(\phi_1)(\zeta, \zeta' \sigma', f), \Phi(\phi_2)(\zeta, \zeta' \sigma', f)) = d(\zeta' \phi_1(\zeta', \sigma', f), \zeta' \phi_2(\zeta', \sigma', f)) \leq \frac{1}{2} \cdot d(\phi_1(\zeta', \sigma', f), \phi_2(\zeta', \sigma', f)) [\text{Example A.16(3)}] \leq \frac{1}{2} \cdot d(\phi_1, \phi_2).
  \]

From Banach’s theorem we can conclude that the contractive function $\Phi$ has a unique fixed point $\text{fix}(\Phi)$: the aimed for semantic construction.
Definition 31. The function $\vdash: \Sigma \times \Sigma^\infty \times (\Sigma \rightarrow \Sigma^\infty) \rightarrow \Sigma^\infty$ is defined by

$\vdash = \text{fix}(\Phi)$.

Note that we have justified the definition we started with, since we can deduce from Banach’s theorem that the semantic sequential composition ‘$;$’ is the unique element of the complete metric space $\Sigma \times \Sigma^\infty \times (\Sigma \rightarrow \Sigma^\infty) \rightarrow \Sigma^\infty$ satisfying

$\sigma;f = \begin{cases} f(\zeta) & \text{if } \sigma = \epsilon, \\ \zeta'(\sigma';\vdash f) & \text{if } \sigma = \zeta'\sigma'. \end{cases}$

This semantic operator has the following property. This property will be exploited in the justification of the definition of the semantic counterpart of the while statement (see Proposition 35).

Proposition 32. For all $\zeta \in \Sigma$, $\sigma \in \Sigma^\infty$, and $f_1, f_2 \in \Sigma \rightarrow \Sigma^\infty$,

$d(\sigma; f_1, \sigma; f_2) \leq \begin{cases} d(f_1, f_2) & \text{if } \sigma = \epsilon, \\ \frac{1}{2} \cdot d(f_1, f_2) & \text{otherwise}. \end{cases}$

Proof. Left as an exercise. □

Exercise 33. Prove Proposition 32. Hint: first prove the following fact and exploits it together with Banach’s theorem. Let $X$ be a metric space and let $(x_n)_n$ and $(y_n)_n$ be converging sequences in $X$. Let $\varepsilon \geq 0$. If $d(x_n, y_n) \leq \varepsilon$ for all $n \in \mathbb{N}$, then $d(\lim_n x_n, \lim_n y_n) \leq \varepsilon$.

As we have already seen in Exercise 24, in the operational semantics we have the following basic equivalence characterising the while statement:

$\text{while } b \text{ do } s \text{ od} = \text{if } b \text{ then } s; \text{while } b \text{ do } s \text{ od} \text{ else skip fi}$.

Also in the denotational semantics we want this equivalence to hold. Consequently,

$\mathcal{D}(\text{while } b \text{ do } s \text{ od})$

$= \mathcal{D}(\text{if } b \text{ then } s; \text{while } b \text{ do } s \text{ od} \text{ else skip fi})$

$= \lambda \zeta. \begin{cases} \mathcal{D}(s; \text{while } b \text{ do } s \text{ od})(\zeta) & \text{if } B(b)(\zeta) = \text{true}, \\ \mathcal{D}(\text{skip})(\zeta) & \text{if } B(b)(\zeta) = \text{false}, \end{cases}$

$= \lambda \zeta. \begin{cases} \mathcal{D}(s)(\zeta); \mathcal{D}(\text{while } b \text{ do } s \text{ od}) & \text{if } B(b)(\zeta) = \text{true}, \\ \zeta & \text{if } B(b)(\zeta) = \text{false}. \end{cases}$

According to the compositionality principle, we have to introduce for $\beta \in \Sigma \rightarrow \{\text{true}, \text{false}\}$ and $f \in \Sigma \rightarrow \Sigma^\infty$ an operation $\psi(\beta, f): \Sigma \rightarrow \Sigma^\infty$ such that

$\mathcal{D}(\text{while } b \text{ do } s \text{ od}) = \psi(B(b), \mathcal{D}(s)).$
From the above we can deduce that this operation should satisfy the recursive equation

\[
\psi(\mathcal{B}(b), \mathcal{D}(s)) = \lambda_{\xi} \left\{ \begin{array}{ll}
\mathcal{D}(s)(\xi) \cdot \psi(\mathcal{B}(b), \mathcal{D}(s)) & \text{if } \mathcal{B}(b)(\xi) = \text{true}, \\
\xi & \text{if } \mathcal{B}(b)(\xi) = \text{false}.
\end{array} \right.
\]  

(3)

Like the semantic sequential composition, also this operation is defined as the unique fixed point of a contractive function from a complete metric space to itself. Again we endow the set \( \Sigma^\infty \) with the Baire metric and the set \( \Sigma \) with the discrete one. From these complete metric spaces we construct the complete space \( \Sigma \rightarrow \Sigma^\infty \). A function from this complete metric space to itself is introduced in

**Definition 34.** Let \( \beta \in \Sigma \rightarrow \{ \text{true}, \text{false} \} \) and \( f \in \Sigma \rightarrow \Sigma^\infty \) such that \( f(\xi) \neq \varepsilon \) for all \( \xi \in \Sigma \). The function \( \Psi(\beta, f) : [\Sigma \rightarrow \Sigma^\infty] \) is defined by

\[
\Psi(\beta, f)(\psi)(\xi) = \left\{ \begin{array}{ll}
f(\xi) \cdot \psi & \text{if } \beta(\xi) = \text{true}, \\
\xi & \text{if } \beta(\xi) = \text{false}.
\end{array} \right.
\]

Next, we show that this function is a contraction.

**Proposition 35.** The function \( \Psi(\beta, f) \) is contractive.

**Proof.** Let \( \psi_1, \psi_2 \in \Sigma \rightarrow \Sigma^\infty \) and \( \xi \in \Sigma \). We distinguish two cases.

- Assume \( \beta(\xi) = \text{true} \). Then

\[
d(\Psi(\beta, f)(\psi_1)(\xi), \Psi(\beta, f)(\psi_2)(\xi)) = d(f(\xi) \cdot \psi_1, f(\xi) \cdot \psi_2)
\]

\[
\leq \frac{1}{2} \cdot d(\psi_1, \psi_2) \quad \text{[Proposition 32]}
\]

- If \( \beta(\xi) = \text{false} \) then

\[
d(\Psi(\beta, f)(\psi_1)(\xi), \Psi(\beta, f)(\psi_2)(\xi)) = d(\xi, \xi)
\]

\[
\leq \frac{1}{2} \cdot d(\psi_1, \psi_2). \quad \Box
\]

According to Banach’s theorem, its unique fixed point \( \text{fix}(\Psi(\beta, f)) \) is the unique function satisfying

\[
\text{fix}(\Psi(\beta, f)) = \lambda_{\xi} \left\{ \begin{array}{ll}
f(\xi) \cdot \text{fix}(\Psi(\beta, f)) & \text{if } \beta(\xi) = \text{true}, \\
\xi & \text{if } \beta(\xi) = \text{false},
\end{array} \right.
\]

(see Eq. (3)). The denotational semantics is now defined as follows.
**Definition 36.** The function $\mathcal{D} : \text{Stat} \to \Sigma \to \Sigma^\infty$ is defined by

\[
\mathcal{D}(v := e) = \lambda_{\xi.} \{ n/v \}, \quad \text{where } n = \mathcal{E}(e)(\xi), \\
\mathcal{D}(\text{skip}) = \lambda_{\xi.}, \\
\mathcal{D}(s_1; s_2) = \lambda_{\xi.} \mathcal{D}(s_1)(\xi) \cdot \mathcal{D}(s_2), \\
\mathcal{D}(\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi}) = \lambda_{\xi.} \begin{cases} \\
\mathcal{D}(s_1)(\xi) & \text{if } \mathcal{B}(b)(\xi) = \text{true}, \\
\mathcal{D}(s_2)(\xi) & \text{if } \mathcal{B}(b)(\xi) = \text{false}, \\
\end{cases} \\
\mathcal{D}(\text{while } b \text{ do } s \text{ od}) = \text{fix}(\Psi(\mathcal{B}(b), \mathcal{D}(s))).
\]

Note that the function $\Psi(\beta, f)$ is only defined for $f \in \Sigma \to \Sigma^\infty$ satisfying for all $\xi \in \Sigma$, $f(\xi) \neq e$. Therefore, we still have to check that for all $s \in \text{Stat}$ and $\xi \in \Sigma$, $\mathcal{D}(s)(\xi) \neq e$. We leave this to the reader.

**Exercise 37.** Prove that for all $s \in \text{Stat}$ and $\xi \in \Sigma$, $\mathcal{D}(s)(\xi) \neq e$.

We conclude the study of the denotational semantics with

**Example 38.** According to Banach’s theorem,

\[
\mathcal{D}(\text{while true do skip od})(\xi) \\
= \text{fix}(\Psi(\mathcal{B}(\text{true}), \mathcal{D}(\text{skip))))(\xi) \\
= \lim_{n} \psi_n(\xi),
\]

where $\psi_0 = \lambda_{\xi.} \xi$ and $\psi_{n+1} = \Psi(\mathcal{B}(\text{true}), \mathcal{D}(\text{skip}))(\psi_n)$. One can easily verify that for all $n \in \mathbb{N}$, $\psi_n = \lambda_{\xi.} \xi^{n+1}$ by induction on $n$. Consequently, we can conclude that $\mathcal{D}(\text{while true skip od})(\xi) = \xi^0$.

### 3.4. Relating $\mathcal{C}$ and $\mathcal{D}$

Unlike in Section 2.4, we cannot prove the operational and denotational semantics to coincide by structural induction. We can deduce that

\[
\mathcal{C}(\text{while } b \text{ do } s \text{ od})(\xi) \\
= \mathcal{C}(\text{if } b \text{ then } s; \text{ while } b \text{ do } s \text{ od} \text{ else } \text{skip fi})(\xi) \quad \text{[Exercise 24]} \\
= \begin{cases} \\
\mathcal{C}(s; \text{ while } b \text{ do } s \text{ od})(\xi) & \text{if } \mathcal{B}(b)(\xi) = \text{true}, \\
\mathcal{C}(\text{skip})(\xi) & \text{if } \mathcal{B}(b)(\xi) = \text{false}. \\
\end{cases}
\]

Since $s; \text{ while } b \text{ do } s \text{ od}$ is structurally more complex than $\text{while } b \text{ do } s \text{ od}$, we cannot apply the induction hypothesis (if we were to use structural induction) at this point. Notice however that from the above we can almost conclude that $\mathcal{C}(\text{while } b \text{ do } s \text{ od})$ satisfies Eq. (3), which uniquely defines $\mathcal{D}(\text{while } b \text{ do } s \text{ od})$. If we could actually
prove this, then we could deduce from Banach’s theorem that the two semantic models coincide for the while statement.

To prove the coincidence of the models we exploit a more generally applicable proof principle based on Banach’s theorem. The unique fixed point proof principle is exploited to prove elements of a metric space to be equal by introducing a function from the metric space to itself, verifying that this function is contractive, and checking that the elements are a fixed point of the function – this fixed point is unique by Banach’s theorem. To apply the proof principle to this setting, we first observe that both the operational and the denotational semantics are an element of the metric space $\Sigma \to \Sigma^\infty$.

Next, we have to introduce a function from this metric space to itself. This function transforms semantic models into semantic models and is therefore called a semantics transformation. Here we exploit the theory developed in Section B.3.1: given a labelled transition system, a corresponding semantics transformation is introduced, this function is shown to be a contractive function from a metric space to itself, and the operational semantics induced by the labelled transition system is proved to be a fixed point of the semantics transformation. So we only have left to show that also the denotational semantics is a fixed point of the semantics transformation.

Definition B.22 amounts in this case to the following. The semantics transformation $T : [\text{Stat}_E \times \Sigma \to \Sigma^\infty]$ is given by

$$T([\delta, z]) = \begin{cases} e & \text{if } [\delta, z] \not\rightarrow, \\ \xi' T([\delta', z']) & \text{if } [\delta, z] \xrightarrow{\xi'} [\delta', z']. \end{cases}$$

Like in Section 2.4, we extend the denotational semantics from statements to configurations.

**Definition 39.** The function $\mathcal{D} : \text{Stat}_E \times \Sigma \to \Sigma^\infty$ is defined by

$$\mathcal{D}([e, z]) = e, \quad \mathcal{D}([s, z]) = \mathcal{D}(s)(z).$$

The fact that the extended denotational semantics $\mathcal{D}$ is a fixed point of the semantics transformation $T$ is proved by induction on the complexity of the statements. This complexity function is defined as follows (see the proof of Proposition 9).

**Definition 40.** The function $\text{comp} : \text{Stat}_E \to \mathbb{N}$ is defined by

$$\begin{align*}
\text{comp}(e) & = 0, \\
\text{comp}(v := e) & = 1, \\
\text{comp}(\text{skip}) & = 1, \\
\text{comp}(s_1; s_2) & = \text{comp}(s_1) + 1, \\
\text{comp}(\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi}) & = \max\{\text{comp}(s_1), \text{comp}(s_2)\} + 1, \\
\text{comp}(\text{while } b \text{ do } s \text{ od}) & = \text{comp}(s) + 3.
\end{align*}$$
This complexity function is defined in such a way that
\[
\text{comp}(\text{if } b \text{ then } s \text{ while } b \text{ do } s \text{ od else skip fi}) = \max\{\text{comp}(s \text{ while } b \text{ do } s \text{ od}), \text{comp}(\text{skip})\} + 1
\]
\[
= \text{comp}(s) + 1 + 1
\]
\[
< \text{comp}(s) + 3
\]
\[
= \text{comp}(\text{while } b \text{ do } s \text{ od}).
\]

We will exploit this in the final case of the next proof. The natural number \(\text{comp}(\bar{s})\) gives us an upper bound of the height of any proof of a transition starting from \(\bar{s}\) (even if we replace (5) of Definition 23 by (6) of Exercise 24).

**Theorem 41.** \(\mathcal{F}(\mathcal{D}) = \mathcal{D}\).

**Proof.** We prove that for all \(\bar{s} \in \text{Stat}_e\) and \(\zeta \in \Sigma\),
\[
\mathcal{F}(\mathcal{D})([\bar{s}, \zeta]) = \mathcal{D}([\bar{s}, \zeta])
\]
by induction on \(\text{comp}(\bar{s})\). We distinguish the following cases.

- Let \(\bar{s} = e\). Then

  \[
  \mathcal{F}(\mathcal{D})([e, \zeta])
  = e
  = \mathcal{D}([e, \zeta]).
  \]

- Let \(\bar{s} = v := e\) and assume \(n = \delta(e)(\zeta)\). Then

  \[
  \mathcal{F}(\mathcal{D})([v := e, \zeta])
  = \zeta\{n/v\} \mathcal{D}([e, \zeta\{n/v\}])
  = \zeta\{n/v\}
  = \mathcal{D}([v := e, \zeta]).
  \]

- Let \(\bar{s} = \text{skip}\). Similar to the previous case.

- Let \(\bar{s} = s_1 ; s_2\) and assume \([s_1, \zeta] \xrightarrow{\zeta'} [e, \zeta']\). In this case,

  \[
  \mathcal{F}(\mathcal{D})([s_1; s_2, \zeta])
  = \zeta' \mathcal{D}([s_2, \zeta'])
  = \zeta' ; \zeta \mathcal{D}(s_2)
  = \mathcal{F}(\mathcal{D})([s_1, \zeta]) ; \mathcal{D}(s_2)
  = \mathcal{D}([s_1, \zeta]) ; \mathcal{D}(s_2) \quad \text{[induction]}
  = \mathcal{D}([s_1; s_2, \zeta]).
  \]
• Let \( s = s_1 ; s_2 \) and assume \([s_1 \od s_2] \xrightarrow{\cdot} [s'_1 \od s'_2] \). In this case,

\[
\begin{align*}
\mathcal{F}(\mathcal{D})([s_1 \od s_2]) &= \mathcal{F}(\mathcal{D}([s'_1 \od s'_2])) \\
&= \mathcal{F}((\mathcal{D}([s'_1 \od s'_2])); \mathcal{D}(s_2)) \\
&= (\mathcal{D}((\mathcal{D}([s'_1 \od s'_2])); \mathcal{D}(s_2)) \\
&= \mathcal{F}(\mathcal{D}([s_1 \od])) \od \mathcal{D}(s_2) \\
&= \mathcal{D}([s_1 \od]) \od \mathcal{D}(s_2) \quad \text{[induction]} \\
&= \mathcal{D}([s_1 \od s_2] \od).
\end{align*}
\]

• If \( s = \text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi and } B(b)(\od) = \text{true} \), then

\[
\begin{align*}
\mathcal{F}(\mathcal{D})([\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi, } \od]) &= \mathcal{F}(\mathcal{D}([s_1 \od])) \\
&= \mathcal{D}([s_1 \od]) \quad \text{[induction]} \\
&= \mathcal{D}([\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi, } \od]).
\end{align*}
\]

• Let \( s = \text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi and } B(b)(\od) = \text{false} \). Similar to the previous case.

• If \( s = \text{while } b \text{ do } s \text{ od} \) then

\[
\begin{align*}
\mathcal{F}(\mathcal{D})([\text{while } b \text{ do } s \text{ od, } \od]) &= \mathcal{F}(\mathcal{D})([\text{if } b \text{ then } s \text{; while } b \text{ do } s \text{ od else skip fi, } \od]) \quad \text{[Exercise 24]} \\
&= \mathcal{D}([\text{if } b \text{ then } s \text{; while } b \text{ do } s \text{ od else skip fi, } \od]) \quad \text{[induction]} \\
&= \lambda\od \begin{cases} 
\mathcal{D}(s \text{; while } b \text{ do } s \od)(\od) & \text{if } B(b)(\od) = \text{true}, \\
\mathcal{D}((\text{skip})(\od)) & \text{if } B(b)(\od) = \text{false}, \\
\od & \text{if } B(b)(\od) = \text{false}, 
\end{cases} \\
&= \lambda\od \begin{cases} 
\mathcal{D}(s)(\od) \od \mathcal{D}(\text{while } b \text{ do } s \od) & \text{if } B(b)(\od) = \text{true}, \\
\od & \text{if } B(b)(\od) = \text{false}, 
\end{cases} \\
&= \mathcal{D}([\text{while } b \text{ do } s \od] \od).
\]

Having shown that the denotational semantics is also a fixed point of the semantics transformation, we arrive by uniqueness of fixed point at

**Corollary 42.** \( \mathcal{C} = \mathcal{D} \).

**Proof.** For all \( s \in \text{Stat} \) and \( \od \in \Sigma \),

\[
\mathcal{C}(s)(\od) = \mathcal{C}([s, \od])
\]
4. Bounded nondeterminism and nontermination

Again we add a programming construct to the language. This time the language is augmented with the \textit{parallel composition}. As we will see, this construction introduces \textit{nondeterminism}: not all steps in a computation are fully determined, that is, at some points in the computation there is a choice which step to do next.

Statements composed in parallel compute independently. They can exchange information by the variables they share. We model the execution of the parallel composition $s_1 \parallel s_2$ by interleaving the executions of the statements $s_1$ and $s_2$. For example, the execution of

$$(v := 1; v := v + 1) \parallel v := 2$$

gives rise to the execution of one of the following sequences of basic statements:

- $v := 1, \; v := v + 1, \; v := 2$,
- $v := 1, \; v := 2, \; v := v + 1$,
- $v := 2, \; v := 1, \; v := v + 1$.

Note that in every sequence $v := 1$ precedes $v := v + 1$.

The parallel composition is modelled operationally by adding four new rules. As we will see, the new collection of axioms and rules gives rise to a \textit{nondeterministic} labelled transition system: some configurations have multiple outgoing transitions. However, the amount of nondeterminism we encounter in the system is \textit{bounded}. Every configuration has only \textit{finitely} many outgoing transitions. Therefore, we call the system \textit{finitely branching}. The finitely branchingness of the system will be crucial when we relate the operational semantics to a denotational one. Because the labelled transition system is nondeterministic, the operational semantics assigns to each statement a function mapping each state to a set of state sequences. The presence of the parallel composition also causes that the operational semantics lacks compositionality. As we will observe, in general we cannot determine the operational semantics of a parallel composition $s_1 \parallel s_2$ from the operational semantics of $s_1$ and $s_2$. For the language at hand, the functions from states to sets of state sequences, as used in the operational semantics, do not contain enough information about the computations to give a compositional semantics. Instead, in the denotational semantics we use tree-like entities – these are called \textit{processes} from now on. Processes were already introduced by De Bakker and Zucker in [16]. Like in Section 3, we endow this mathematical structure with a metric. The metric space of processes is defined as the unique solution of a \textit{recursive equation} over complete metric spaces. In Appendix D, we sketch how to build such recursive equations and we specify which equations have unique solutions. A key ingredient in the equation defining the metric space of processes is the following construction. Given
a metric space $X$, take the set $\mathcal{P}_{\text{nc}}(X)$ of nonempty and \textit{compact} — compactness being a natural generalisation of finiteness — subsets of $X$ and endow them with the induced \textit{Hausdorff metric}. The restriction to compact sets is crucial. If we were to consider all nonempty subsets, then we would not obtain a metric space in general.

In this section, the operational semantics $\mathcal{O}$ and denotational semantics $\mathcal{D}$ do not coincide. We relate the two models by means of a \textit{linearise operator} $\text{lin}$ which abstracts from the additional information present in the denotational semantics. An operator similar to $\text{lin}$ was already studied by De Bakker et al. in [11]. We will prove

$$\mathcal{O} = \text{lin} \circ \mathcal{D}.$$ 

To prove this result we introduce an \textit{intermediate semantics} $\mathcal{I}$. Like the denotational semantics, the intermediate semantics assigns to each statement a process and in its definition we exploit fixed points. Like the operational semantics, the definition of the intermediate semantics is based on a labelled transition system. Kok and Rutten used this type of definition in [34]. The proof of the result stated above is divided into two parts. First, the denotational semantics and the intermediate semantics are shown to be equal. This is proved by uniqueness of fixed point. Second, the intermediate semantics is related to the operational semantics by means of the linearise operator. Again, we exploit the unique fixed point proof principle.

In Section 4.1, we introduce the language. For this language, an operational and a denotational semantics are developed in Section 4.2 and 4.3. The two are linked in Section 4.4.

\subsection*{4.1. Language definition}

To the language defined in Definition 22 we add the parallel composition.

\begin{definition}
The set $(s \in) \textit{Stat of statements}$ is defined by

$$s ::= v := e | \text{skip} | s; s | \text{if } b \text{ then } s \text{ else } s \text{ fi } | \text{while } b \text{ do } s \text{ od } | s || s.$$ 

\end{definition}

For example, the statement

$$v := 0; (\text{while } v \triangleq 100 \text{ do } v := v + 1 \text{ od } || w := v)$$

assigns an arbitrary number between 0 and 100 to the variable $w$.

\subsection*{4.2. Operational semantics}

Again the operational semantics is defined by means of a labelled transition system. Like in Section 3.2, the labelled transition system is nonterminating. This time, the system is nondeterministic. We will show that the labelled transition system at hand is finitely branching.

The configurations and the actions of the labelled transition system are defined as in Section 2.2. The transition relation is extended by adding some rules for the parallel composition.
**Definition 44.** The transition relation \( \rightarrow \) is defined by the following axioms and rules.

1. \([v := e, \xi] \xrightarrow{\xi(n/v)} [E, \xi\{n/v\}]\), where \( n = \mathcal{E}(\xi) \).

2. \([\text{skip}, \xi] \xrightarrow{\xi}[E, \xi]\),

3. \[\begin{align*}
[s_1, \xi] & \xrightarrow{\xi}[E, \xi'] \quad \quad [s_1, \xi] \xrightarrow{\xi}[s_1', \xi'] \\
[s_1 \parallel s_2, \xi] & \xrightarrow{\xi}[s_2, \xi'] \\
[s_1; s_2, \xi] & \xrightarrow{\xi}[s_1'; s_2, \xi']
\end{align*}\]

4. \[\begin{align*}
[s_1, \xi] & \xrightarrow{\xi}[\tilde{s}_1, \xi'] \\
[s_2, \xi] & \xrightarrow{\xi}[\tilde{s}_2, \xi']
\end{align*}\] if \( \mathcal{A}(b)(\xi) = \text{true} \),

5. \[\begin{align*}
[s, \xi] & \xrightarrow{\xi}[E, \xi'] \\
[s] & \xrightarrow{\xi}[s', \xi']
\end{align*}\] if \( \mathcal{A}(b)(\xi) = \text{true} \),

6. \[\begin{align*}
[s_1, \xi] & \xrightarrow{\xi}[E, \xi'] \\
[s_1 \parallel s_2, \xi] & \xrightarrow{\xi}[s_1', \xi'] \\
[s_1; s_2, \xi] & \xrightarrow{\xi}[s_2', \xi']
\end{align*}\]

All but the rules for the parallel composition have already been discussed before. The statements \( s_1 \) and \( s_2 \) composed in parallel compute independently. This is modelled by interleaving their transitions as described by the above rules. If one of the two statements terminates, the execution continues with the other remaining statement. The execution of the parallel composition terminates precisely when both statements have terminated.

The labelled transition system is nondeterministic as is shown in

**Example 45.** Since

\[\begin{align*}
[v := 1, \xi] & \xrightarrow{\xi(1/v)} [E, \xi\{1/v\}] \\
[v := 1; v := 2, \xi] & \xrightarrow{\xi(1/v)} [v := 2, \xi\{1/v\}]
\end{align*}\]
and

\[
[v := 3, \varsigma] \xrightarrow{\{3/v\}} [E, \varsigma\{3/v\}]
\]

\[
[v := 1; v := 2 \parallel v := 3, \varsigma] \xrightarrow{\{3/v\}} [v := 1; v := 2, \varsigma\{3/v\}]
\]

the labelled transition system is nondeterministic.

Every configuration has only finitely many outgoing transitions.

**Proposition 46.** The labelled transition system is finitely branching.

**Proof.** We prove that for all \(s \in \text{Stat}_E\) and \(\varsigma \in \Sigma\), the set

\[
\mathcal{S}([s, \varsigma]) = \{ \langle s', [s', \varsigma'] \rangle \mid [s, \varsigma] \xrightarrow{\mathcal{S}} [s', \varsigma'] \}
\]

is finite by structural induction on \(s\). The details are left to the reader. \(\square\)

**Exercise 47.** Complete the proof of Proposition 46.

According to Definition B.16, the nondeterministic and nonterminating labelled transition system defines an operational semantics

\(\mathcal{O} : \text{Stat}_E \times \Sigma \rightarrow \mathcal{P}_n(\Sigma^\infty)\)

given by

\[
\mathcal{O}([s, \varsigma]) = \{ \varsigma_1 \varsigma_2 \cdots \varsigma_n \mid [s, \varsigma] = [s_0, \varsigma_0] \xrightarrow{\varsigma_1} [s_1, \varsigma_1] \xrightarrow{\varsigma_2} \cdots \xrightarrow{\varsigma_n} [s_n, \varsigma_n] \} \cup \\
\{ \varsigma_1 \varsigma_2 \cdots \varsigma_n \mid [s, \varsigma] = [s_0, \varsigma_0] \xrightarrow{\varsigma_1} [s_1, \varsigma_1] \xrightarrow{\varsigma_2} \cdots \}
\]

**Definition 48.** The function \(\mathcal{O} : \text{Stat} \rightarrow \Sigma \rightarrow \mathcal{P}_n(\Sigma^\infty)\) is defined by

\(\mathcal{O}(s) = \lambda \varsigma.\mathcal{O}([s, \varsigma]).\)

**Exercise 49.** Prove that the operational semantics is not compositional.

### 4.3. Denotational semantics

We need a space containing more structure than the functions from states to sets of state sequences used in the operational semantics to give a denotational semantics. In Definition 50 below we introduce the complete metric space \(\mathcal{P}\). The elements of this space can be viewed as tree-like objects. It will turn out that this structure is rich enough to model the parallel composition (and the other constructs) compositionally.

**Definition 50.** The complete metric space \((p \in) \mathcal{P}\) of processes is defined as the solution of the recursive equation

\[
\mathcal{P} \cong (\Sigma \rightarrow \mathcal{P}_n(\Sigma \times \frac{1}{2} \cdot \mathcal{P})) + \{ \sqrt{ } \}.
\]
In the above definition, the set $\Sigma$ is assumed to be endowed with the discrete metric and the set $\{\sqrt{\cdot}\}$ with the obvious one. Since every function from a set endowed with the discrete metric – like $\Sigma$ in the above recursive equation – to some other metric space is nonexpansive, we can replace the $\rightarrow$ by $\overset{\rightarrow}{\rightarrow}$ in the above definition. According to Theorem D.3, the above recursive equation has indeed a unique solution.

Processes can be viewed as tree-like objects. We distinguish the following two cases.

- $p = \sqrt{\cdot}$: We use $\sqrt{\cdot}$ to model successful termination in the denotational semantics, like we exploit $\varepsilon$ in the operational semantics to handle termination. It can be seen as the empty tree consisting of one node and no edges.
- $p \neq \sqrt{\cdot}$: Let, for $\zeta, \zeta' \in \Sigma$,

$$
p(\zeta) = \{\langle \zeta_1, p_1 \rangle, \ldots, \langle \zeta_m, p_m \rangle\}
$$

$$
p(\zeta') = \{\langle \zeta'_1, p'_1 \rangle, \ldots, \langle \zeta'_n, p'_n \rangle, \ldots\}.
$$

The process $p$ can be viewed as the labelled tree

In the above picture the upper level of branching is due to the functional nature of $p$. It records the change of the state caused by the environment (to be thought of as a process in parallel). The lower level of branching stems from the set structure of $p(\zeta)$ and $p(\zeta')$. It records the nondeterminism caused by the parallel composition. For example, if the environment changes the state to $\zeta$ then the process $p$ can change the state to $\zeta_1, \ldots, \zeta_m$ followed by $p_1, \ldots, p_m$, respectively.

These labelled trees can be of finite but also of infinite depth.

**Example 51.** Let $\zeta \in \Sigma$. Consider the processes

$$
p_n = \begin{cases} 
\sqrt{\cdot} & \text{if } n = 0, \\
\lambda_{\zeta'}\{\langle \zeta, p_{n-1} \rangle\} & \text{if } n > 0.
\end{cases}
$$

Next we verify that

$$d(p_m, p_n) = \begin{cases} 
0 & \text{if } m = n, \\
2^{-\min\{m,n\}} & \text{otherwise},
\end{cases}
$$

by induction on $\min\{m,n\}$. We distinguish the following four cases.

- If $m = n$ the above is vacuously true.
• If $m = 0$ and $n > 0$ then
  \[ d_P(p_m, p_n) = d_P(\sqrt{}, p_n) = 1 \quad [p_n \neq \sqrt{}] \]
• The case that $m > 0$ and $n = 0$ is similar to the previous one.
• Let $m, n > 0$ and $m \neq n$. Then
  \[
  d_P(p_m, p_n) = d_P(\lambda_{\xi'} \cdot \{\langle \xi, p_{m-1} \rangle\}, \lambda_{\xi'} \cdot \{\langle \xi, p_{n-1} \rangle\})
  = d_{\Sigma - \mathcal{P}_n(\Sigma \times \frac{1}{2} P)}(\sqrt{}) (\lambda_{\xi'} \cdot \{\langle \xi, p_{m-1} \rangle\}, \lambda_{\xi'} \cdot \{\langle \xi, p_{n-1} \rangle\})
  = d_{\Sigma - \mathcal{P}_n(\Sigma \times \frac{1}{2} P)}(\lambda_{\xi'} \cdot \{\langle \xi, p_{m-1} \rangle\}, \lambda_{\xi'} \cdot \{\langle \xi, p_{n-1} \rangle\})
  = \sup_{\xi' \in \Sigma} d_{\mathcal{P}_n(\Sigma \times \frac{1}{2} P)}(\{\langle \xi, p_{m-1} \rangle\}, \{\langle \xi, p_{n-1} \rangle\})
  = d_{\Sigma \times \frac{1}{2} P}(\langle \xi, p_{m-1} \rangle, \langle \xi, p_{n-1} \rangle)
  = \max\{d_{\xi}(\xi', \xi), \frac{1}{2}, d_P(p_{m-1}, p_{n-1})\}
  = \frac{1}{2} \cdot 2^{-\min\{m-1, n-1\}} \quad [\text{induction}]
  = 2^{-\min\{m, n\}}.
\]
Consequently, the sequence $(p_n)_n$ is Cauchy. Since the space $\mathcal{P}$ is complete, its limit $\lim_n p_n$ exists. We take the process $p$ to be
\[
\lambda_{\xi'} \cdot \{\langle \xi, p_n \rangle \mid n \in \mathbb{N}\} \cup \{\langle \xi, \lim_n p_n \rangle\}.
\]
This process can be viewed as the labelled tree (in the picture below we have left out the levels corresponding to the functional nature of the process, since all the functions are constant)

To the basic constructs of the language $v := e$ and skip we associate the constants
\[
\lambda_{\xi} \cdot \{\langle \xi \{n/v\}, \sqrt{\rangle\}, \quad \text{where} \ n = \delta(e)(\xi)
\]
and

$$\lambda \zeta \cdot \{ (\zeta, \sqrt{\cdot}) \}.$$

Obviously, the singleton sets $\{ (\zeta \{ n/v \}, \sqrt{\cdot}) \}$ and $\{ (\zeta, \sqrt{\cdot}) \}$ are compact. These processes can be viewed as the labelled trees

To model the if statement denotationally we make use of the following construction. For $\beta \in \Sigma \rightarrow \{ \text{true}, \text{false} \}$ and $p, q \in \mathcal{P}$ we take

$$\lambda \zeta \cdot \begin{cases} p(\zeta) & \text{if } \beta(\zeta) = \text{true}, \\ q(\zeta) & \text{if } \beta(\zeta) = \text{false}. \end{cases}$$

For handling the sequential composition in the denotational setting, we are looking for a semantic operator $\cdot ; : \mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P}$ satisfying

$$p ; q = \begin{cases} q & \text{if } p = \sqrt{\cdot}, \\ \lambda \zeta \cdot \{ (\zeta', p'; q) \mid (\zeta', p') \in p(\zeta) \} & \text{otherwise}. \end{cases}$$

For the justification of the above definition we follow the same route as we did in Section 3.3. We introduce a function $\Phi$ (in Definition 56) from a complete metric space to itself, show that this function is a contraction, and define the semantic sequential composition as its unique fixed point. If we want to verify that $\Phi$ is a function from the complete metric space $\mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P}$ to itself, we have to show that for all $\phi \in \mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P}$, $p, q \in \mathcal{P}$, and $\zeta \in \Sigma$, the set

$$\{ (\zeta', \phi(\zeta', p'; q)) \mid (\zeta', p') \in p(\zeta) \}$$

is compact. However, this is in general not the case as is shown in the following example.

**Example 52.** Let $p$ be the process introduced in Example 51. We take the function $\phi$ to be

$$\lambda (p, q) \cdot \begin{cases} \lambda \zeta \cdot (\zeta_n, \sqrt{\cdot}) & \text{if } p = p_n \text{ for some } n \in \mathbb{N}, \\ \sqrt{\cdot} & \text{otherwise}. \end{cases}$$
Let $q$ be an arbitrary process. The process $\phi(p_n, q)$ can be seen as the labelled tree

```
  .
 / \  \\
ζ_n  .
```

For all $ζ \in Σ$, the set (4) amounts to

$$\{\langle ζ, λζ', \{ζ_n, \sqrt{\}}\rangle | n \in \mathbb{N}\} \cup \{\langle ζ, \sqrt{\}\rangle\}.$$ 

This set corresponds to the process which can be viewed as

```
  .
 / \  \\
ζ  .
```

The set is not compact, since the sequence $(\langle ζ, λζ', \{ζ_n, \sqrt{\})\rangle)_n$ has no converging subsequence.

If we restrict ourselves to nonexpansive $\phi$’s, then we can prove that the set (4) is compact.

**Proposition 53.** For all $\phi \in \mathbb{P} \times \mathbb{P} \rightarrow \mathbb{P}$, $p, q \in \mathbb{P}$, and $ζ \in Σ$, the set

$$\{\langle ζ', \phi(p', q)\rangle | \langle ζ', p'\rangle \in p(ζ)\}$$

is compact.

**Proof.** Let $(\langle ζ'_n, \phi(p'_n, q)\rangle)_n$ be a sequence in the set. Then $(\langle ζ'_n, p'_n\rangle)_n$ is a sequence in $p(ζ)$. Since the set $p(ζ)$ is compact, there exists a subsequence $(\langle ζ'_n, p'_n(ζ)\rangle)_n$ converging to some $\langle ζ', p'\rangle$ in $p(ζ)$. We leave it to the reader to verify that the sequence $(\langle ζ'_n, \phi(p'_n(ζ), q)\rangle)_n$ converges to $\langle ζ', \phi(p', q)\rangle$ (Exercise 54). Clearly, $\langle ζ', \phi(p', q)\rangle$ is in the set. □

**Exercise 54.** Complete the proof of Proposition 53.

Because we have restricted ourselves to nonexpansive $\phi$’s, we should also check that $Φ(\phi)$ is nonexpansive.

**Proposition 55.** For all $\phi \in \mathbb{P} \times \mathbb{P} \rightarrow \mathbb{P}$, $p_1, p_2, q_1, q_2 \in \mathbb{P}$, and $ζ \in Σ$,

$$d(\{\langle ζ'_1, \phi(p'_1, q_1)\rangle | \langle ζ'_1, p'_1\rangle \in p_1(ζ)\} \cup \{\langle ζ'_2, \phi(p'_2, q_2)\rangle | \langle ζ'_2, p'_2\rangle \in p_2(ζ)\})$$

$$\leq \max\{d(p_1, p_2), d(q_1, q_2)\}.$$
Proof. Let $⟨\varsigma'_1, p'_1⟩ \in p_1(\varsigma)$. Then there exists a $⟨\varsigma'_2, p'_2⟩ \in p_2(\varsigma)$ such that
\[
d(⟨\varsigma'_1, p'_1⟩, ⟨\varsigma'_2, p'_2⟩) \leq d(p_1(\varsigma), p_2(\varsigma)) \leq d(p_1, p_2).
\]
Hence, $d(⟨\varsigma'_1, \varsigma'_2⟩) \leq d(p_1, p_2)$ and $d(p'_1, p'_2) \leq 2 \cdot d(p_1, p_2)$. Consequently,
\[
d(⟨\varsigma'_1, \phi(p'_1, q_1)⟩, ⟨\varsigma'_2, \phi(p'_2, q_2)⟩) = \max\{d(⟨\varsigma'_1, \varsigma'_2⟩), \frac{1}{2} \cdot d(\phi(p'_1, q_1), \phi(p'_2, q_2))\} \leq \max\{d(⟨\varsigma'_1, \varsigma'_2⟩), \frac{1}{2} \cdot d(p'_1, q_1), d(p'_2, q_2)\} \quad [\phi \text{ is nonexpansive}]
\]
\[
\leq \max\{d(p_1, p_2), d(q_1, q_2)\}. \quad \square
\]

Now we are ready to introduce the function $ϕ$ from the complete metric space $\mathbb{P} \times \mathbb{P} \longrightarrow \mathbb{P}$ to itself.

Definition 56. The function $ϕ : [\mathbb{P} \times \mathbb{P} \longrightarrow \mathbb{P}]$ is defined by
\[
ϕ(p, q) = \begin{cases} q & \text{if } p = \sqrt{}, \\ λ_ς.\{⟨\varsigma', ϕ(p', q)⟩ \mid ⟨\varsigma', p'⟩ \in p(ς)\} & \text{otherwise}. \end{cases}
\]

To apply Banach’s theorem we have to verify that the function is a contraction.

Proposition 57. The function $ϕ$ is contractive.

Proof. Left to the reader as an exercise. \square

Exercise 58. Prove Proposition 57.

From Banach’s theorem we can conclude that $ϕ$ has a unique fixed point $\text{fix}(ϕ)$.

Definition 59. The function $; : [\mathbb{P} \times \mathbb{P} \longrightarrow \mathbb{P}]$ is defined by
\[
; = \text{fix}(ϕ).
\]

The semantic sequential composition ‘;’ is the unique element of the complete metric space $\mathbb{P} \times \mathbb{P} \longrightarrow \mathbb{P}$ satisfying
\[
p; q = \begin{cases} q & \text{if } p = \sqrt{}, \\ λ_ς.\{⟨\varsigma', p'; q⟩ \mid ⟨\varsigma', p'⟩ \in p(ς)\} & \text{otherwise}. \end{cases}
\]

By definition, this operator is nonexpansive. Furthermore, we have the following property.
Proposition 60. For all $p, q_1, q_2 \in P$,
\[
d(p; q_1, p; q_2) \leq \begin{cases} 
d(q_1, q_2) & \text{if } p = \sqrt{,} \\
\frac{1}{2} \cdot d(q_1, q_2) & \text{otherwise}. \end{cases}
\]

Proof. Similar to the proof of Proposition 32. □

To define the semantic counterpart of the while statement we exploit the same techniques as we did in Section 3.3.

Definition 61. Let $\beta \in \Sigma \to \{true, false\}$ and $p \in P$ such that $p \neq \sqrt{.}$ The function $\Psi(\beta, p) : [P]$ is defined by
\[
\Psi(\beta, p)(\psi) = \lambda z. \left\{ \begin{array}{ll}
(p; \psi)(z) & \text{if } \beta(z) = true, \\
\{\langle z, \sqrt{\rangle}\} & \text{if } \beta(z) = false.
\end{array} \right.
\]

Proposition 62. The function $\Psi(\beta, p)$ is contractive.

Proof. Similar to the proof of Proposition 35, this time exploiting Proposition 60. □

The unique fixedpoint of this contractive function from the complete metric space $P$ to itself is the semantic counterpart of the while statement (see Definition 64).

We have left to introduce the semantic parallel composition.

Definition 63. The function $\parallel : P \times P \to P$ is defined by
\[
p \parallel q = \left\{ \begin{array}{ll}
\sqrt & \text{if } p, q = \sqrt, \\
q & \text{if } p = \sqrt \text{ and } q \neq \sqrt, \\
p & \text{if } p \neq \sqrt \text{ and } q = \sqrt, \\
\lambda z. \{\langle z', p' \parallel q \rangle \mid \langle z', p' \rangle \in p(z)\} \cup \\
\{\langle z', p \parallel q' \rangle \mid \langle z', q' \rangle \in q(z)\} & \text{otherwise.}
\end{array} \right.
\]

The justification of the above definition is similar to the one of the semantic sequential composition.

The denotational semantics is given in

Definition 64. The function $\mathcal{D} : Stat \to P$ is defined by
\[
\mathcal{D}(v := e) = \lambda z. \{\langle z[n/v], \sqrt{\rangle\} \text{ where } n = \delta(e)(z), \\
\mathcal{D}\text{(skip)} = \lambda z. \{\langle z, \sqrt{\rangle\}, \\
\mathcal{D}(s_1 ; s_2) = \mathcal{D}(s_1) ; \mathcal{D}(s_2),
\]


D(if b then s₁ else s₂ fi) = \begin{cases} D(s₁)(ζ) & \text{if } \mathcal{B}(b)(ζ) = \text{true}, \\ D(s₂)(ζ) & \text{if } \mathcal{B}(b)(ζ) = \text{false}, \end{cases}

D(while b do s od) = \text{fix}(Ψ(\mathcal{B}(b), D(s))),

D(s₁ || s₂) = D(s₁) || D(s₂).

Since the function Ψ(β, p) is only defined for p ≠ √ we still have to verify that for all s ∈ Stat, D(s) ≠ √. This can be proved similar to Exercise 37.

4.4. Relating \textit{O} and \textit{D}

As we already mentioned in the introduction of Section 4, we cannot expect the operational and denotational semantics to be equal as was the case in the previous sections. We relate the models by means of a linearise operator \textit{lin} introduced in Definition 75 below. This operator assigns to each process \textit{p} and state \textit{ζ} a corresponding set of state sequences. In the rest of this section we prove that

\[ \textit{O} = \textit{lin} \circ \textit{D}. \]

To prove this result we introduce an intermediate semantics \textit{I} in Section 4.4.1. Like the denotational semantics, the intermediate semantics assigns to each statement a process and in its definition we exploit fixed points. Like the operational semantics, the definition of the intermediate semantics is based on the labelled transition system introduced in Definition 44.

The proof of the result stated above is divided into two parts. In Section 4.4.2, the denotational semantics and the intermediate semantics are shown to be equal. The intermediate semantics is related to the operational semantics by means of the linearise operator in Section 4.4.3.

4.4.1. Intermediate semantics

The intermediate semantics has operational and denotational characteristics. It is operational in that it is defined in terms of a labelled transition system and it is denotational in that it exploits fixed points in its definition.

From the labelled transition system introduced in Definition 44 we derive the intermediate semantics \textit{I} : \textit{Stat}_k → \mathbb{P} satisfying

\[ \textit{I}(\textit{OS}'' s) = \begin{cases} √ & \text{if } \textit{OS}'' s = \textit{E}, \\ \lambdaζ' \{ \langle ζ', \textit{I}(\textit{OS}'' s) \rangle \mid [\textit{OS}'' s; ζ] \overset{ζ'}{⇒} [\textit{OS}'' s; ζ'] \} & \text{otherwise}. \end{cases} \]

To justify the above recursive definition we introduce the following function.

\textbf{Definition 65.} The function \( \Omega : [\text{Stat}_k \rightarrow \mathbb{P}] \) is defined by

\[ \Omega(ω)(\textit{̄s}) = \begin{cases} √ & \text{if } \textit{̄s} = \textit{E}, \\ \lambdaζ' \{ \langle ζ', ω(\textit{̄s}) \rangle \mid [\textit{̄s}; ζ] \overset{ζ'}{⇒} [\textit{̄s}; ζ'] \} & \text{otherwise}. \end{cases} \]
Note that since the labelled transition system is finitely branching (Proposition 46), we can conclude that for all \( \bar{s} \in \text{Stat}_E \) and \( \bar{z} \in \Sigma \), the set
\[
\{ (\bar{z'}, \omega(\bar{s})) \mid [\bar{s}, \bar{z}] \xrightarrow{\omega} [\bar{s'}, \bar{z'}] \}
\]
is finite and hence compact. The above introduced function is a contraction.

**Proposition 66.** The function \( \Omega \) is contractive.

**Proof.** Similar to the proof of Proposition 57. \( \square \)

According to Banach’s theorem, the function has a unique fixed point: the intermediate semantics.

**Definition 67.** The function \( \mathcal{I} : \text{Stat}_E \rightarrow \mathbb{P} \) is defined by
\[
\mathcal{I} = \text{fix}(\Omega).
\]

### 4.4.2. Relating \( \mathcal{I} \) and \( \mathcal{D} \)

The intermediate semantics is shown to be equal to the denotational semantics by uniqueness of fixed point. This is done by proving that (a minor extension of) the denotational semantics is also a fixed point of \( \Omega \). We extend the denotational semantics by defining
\[
\mathcal{D}(E) = \sqrt{.}
\]
To prove that this extended denotational semantics is a fixed point of \( \Omega \), we also enhance the complexity function of Definition 40 as follows.

**Definition 68.** The function \( \text{comp} : \text{Stat}_E \rightarrow \mathbb{N} \) is defined by
\[
\begin{align*}
\text{comp}(E) &= 0, \\
\text{comp}(v := e) &= 1, \\
\text{comp}(& \text{skip}) = 1, \\
\text{comp}(s_1 ; s_2) &= \text{comp}(s_1) + 1, \\
\text{comp}(\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi}) &= \max\{\text{comp}(s_1), \text{comp}(s_2)\} + 1, \\
\text{comp}(\text{while } b \text{ do } s \text{ od}) &= \text{comp}(s) + 3, \\
\text{comp}(s_1 \parallel s_2) &= \max\{\text{comp}(s_1), \text{comp}(s_2)\} + 1.
\end{align*}
\]

Now we are ready for

**Theorem 69.** \( \Omega(\mathcal{D}) = \mathcal{D} \).
Proof. We prove that for all $\bar{s} \in \text{Stat} \epsilon$ and $\zeta \in \Sigma$,

$$\Omega(\mathcal{D})(\bar{s})(\zeta) = \mathcal{D}(\bar{s})(\zeta)$$

by induction on $\text{comp}(\bar{s})$. We only consider a few cases.

- Let $\bar{s} = \text{skip}$.

$$\Omega(\mathcal{D})(\text{skip})(\zeta)$$
$$= \{\langle \zeta, \mathcal{D}(E) \rangle \}$$
$$= \{\langle \zeta, \sqrt{\text{}{}} \rangle \}$$
$$= \mathcal{D}(\text{skip})(\zeta).$$

- If $\bar{s} = s_1 ; s_2$ then

$$\Omega(\mathcal{D})(s_1 ; s_2)(\zeta)$$
$$= \{\langle \zeta', \mathcal{D}(s_2) \rangle \mid [s_1, \zeta] \xrightarrow{\cdot} [E, \zeta'] \} \cup$$
$$\{\langle \zeta', \mathcal{D}(s_1 ; s_2) \rangle \mid [s_1, \zeta] \xrightarrow{\cdot} [s_1', \zeta'] \}$$
$$= \{\langle \zeta', \mathcal{D}(E) ; \mathcal{D}(s_2) \rangle \mid [s_1, \zeta] \xrightarrow{\cdot} [E, \zeta'] \} \cup$$
$$\{\langle \zeta', \mathcal{D}(s_1) ; \mathcal{D}(s_2) \rangle \mid [s_1, \zeta] \xrightarrow{\cdot} [s_1', \zeta'] \}$$
$$= \{\langle \zeta', \mathcal{D}(s_1) ; \mathcal{D}(s_2) \rangle \mid [s_1, \zeta] \xrightarrow{\cdot} [s_1', \zeta'] \}$$
$$= (\lambda \zeta.\{\langle \zeta', \mathcal{D}(s_1) \rangle \mid [s_1, \zeta] \xrightarrow{\cdot} [s_1', \zeta'] \} ; \mathcal{D}(s_2))(\zeta)$$
$$= (\Omega(\mathcal{D})(s_1) ; \mathcal{D}(s_2))(\zeta) \quad \text{[induction]}$$
$$= \mathcal{D}(s_1 ; s_2)(\zeta).$$

- If $\bar{s} = \text{while } b \text{ do } s \text{ od}$ then

$$\Omega(\mathcal{D})(\text{while } b \text{ do } s \text{ od})(\zeta)$$
$$= \Omega(\mathcal{D})(\text{if } b \text{ then } s \text{ ; while } b \text{ do } s \text{ od } \text{ else } \text{ skip fi})(\zeta) \quad \text{[Exercise 24]}$$
$$= \mathcal{D}(\text{if } b \text{ then } s \text{ ; while } b \text{ do } s \text{ od } \text{ else } \text{ skip fi})(\zeta) \quad \text{[induction]}$$
$$= \lambda \zeta.\left\{ \begin{array}{ll}
\mathcal{D}(s) & \text{if } B(b)(\zeta) = \text{true} \\
\mathcal{D}(\text{skip})(\zeta) & \text{if } B(b)(\zeta) = \text{false}
\end{array} \right.$$
$$= \lambda \zeta.\left\{ \begin{array}{ll}
(\mathcal{D}(s) ; \mathcal{D}(\text{while } b \text{ do } s \text{ od}))(\zeta) & \text{if } B(b)(\zeta) = \text{true} \\
\{\langle \zeta, \sqrt{\text{}{}} \rangle \} & \text{if } B(b)(\zeta) = \text{false}
\end{array} \right.$$
$$= \mathcal{D}(\text{while } b \text{ do } s \text{ od})(\zeta). \quad \square$$

Exercise 70. Prove the case $\bar{s} = s_1 \parallel s_2$ of Theorem 69.
By uniqueness of fixed point we can conclude

**Corollary 71.** $\mathcal{I} = \mathcal{D}.$

**Proof.** Immediate consequence of Definition 67, Theorem 69, and Banach’s theorem. □

### 4.4.3. Relating $\mathcal{C}$ and $\mathcal{I}$

Next, we relate the operational semantics and the intermediate semantics. For that purpose we introduce the already mentioned linearise operator

$$\text{lin}: \mathbb{P} \rightarrow \Sigma \rightarrow \mathcal{P}_n(\Sigma^\infty).$$

This operator removes the state changes caused by the environment and collapses the branching structure. We define the linearise operator as the unique fixed point of a contractive function from a complete metric space to itself. As usual we restrict ourselves to the subspace $\mathcal{P}_n(\Sigma^\infty)$ of $\mathcal{P}(\Sigma^\infty).$ Like in Definition 56, we only consider nonexpansive functions.

**Definition 72.** The function $\Theta: [\mathbb{P} \rightarrow \Sigma \rightarrow \mathcal{P}_n(\Sigma^\infty)]$ is defined by

$$\Theta(\theta)(p)(\zeta) = \begin{cases} \{\varepsilon\} & \text{if } p = \sqrt{1}, \\
\bigcup \{\varepsilon' \theta(p')(\zeta') | \langle \zeta', p' \rangle \in p(\zeta)\} & \text{otherwise.} \end{cases}$$

In the above definition we use $\varepsilon' \theta(p')(\zeta')$ to denote the set of state sequences $\{\zeta' \sigma | \sigma \in \theta(p')(\zeta')\}.$ We have to verify that for all $\theta \in \mathbb{P} \rightarrow \Sigma \rightarrow \mathcal{P}_n(\Sigma^\infty),$

- for all $p \in \mathbb{P},$ with $p \neq \sqrt{1},$ and $\zeta \in \Sigma,$ the set $\bigcup \{\varepsilon' \theta(p')(\zeta') | \langle \zeta', p' \rangle \in p(\zeta)\}$ is compact, and
- the function $\Theta(\theta)$ is nonexpansive.

We prove these facts in the following two propositions.

**Proposition 73.** For all $\theta \in \mathbb{P} \rightarrow \Sigma \rightarrow \mathcal{P}_n(\Sigma^\infty), p \in \mathbb{P},$ with $p \neq \sqrt{1},$ and $\zeta \in \Sigma,$ the set

$$\bigcup \{\varepsilon' \theta(p')(\zeta') | \langle \zeta', p' \rangle \in p(\zeta)\}$$

is compact.

**Proof.** For all $\langle \zeta', p' \rangle \in p(\zeta),$ the set $\theta(p')(\zeta')$ is compact. From this one can easily deduce that also the set $\varepsilon' \theta(p')(\zeta')$ is compact.

Since the function $\theta$ is nonexpansive, the function $\lambda \langle \zeta', p' \rangle. \varepsilon' \theta(p')(\zeta')$ is also nonexpansive. Because the set $p(\zeta)$ is compact, we can conclude from Alexandroff’s theorem (Theorem A.31) that the set $\{\varepsilon' \theta(p')(\zeta') | \langle \zeta', p' \rangle \in p(\zeta)\}$ is also compact.
Having shown that the set \( \{ \zeta' \theta(p')(\zeta') \mid \langle \zeta', p' \rangle \in p(\zeta) \} \) is a compact set of compact sets we can conclude from Michael’s theorem (Theorem A.28(1)) that its union \( \bigcup \{ \zeta' \theta(p')(\zeta') \mid \langle \zeta', p' \rangle \in p(\zeta) \} \) is also compact.

**Proposition 74.** For all \( \Theta(\theta) \in \mathcal{P} \rightarrow \mathcal{I}_1 \rightarrow \mathcal{P}_\infty(\Sigma^\infty) \), the function \( \Theta(\theta) \) is nonexpansive.

**Proof.** Let \( p_1, p_2 \in \mathcal{P} \) and \( \zeta \in \Sigma \). For all \( \langle \zeta_1', p_1' \rangle \in p_1(\zeta) \), there exists a \( \langle \zeta_2', p_2' \rangle \in p_2(\zeta) \) such that

\[
d(\langle \zeta_1', p_1' \rangle, \langle \zeta_2', p_2' \rangle) \\
\leq d(p_1(\zeta), p_2(\zeta)) \\
\leq d(p_1, p_2).
\]

Because the function \( \lambda \langle \zeta', p' \rangle. \zeta' \theta(p')(\zeta') \) is nonexpansive,

\[
d(\zeta_1' \theta(p_1')(\zeta_1'), \zeta_2' \theta(p_2')(\zeta_2')) \leq d(p_1, p_2).
\]

Hence,

\[
d(\{ \zeta_1' \theta(p_1')(\zeta_1') \mid \langle \zeta_1', p_1' \rangle \in p_1(\zeta) \}, \{ \zeta_2' \theta(p_2')(\zeta_2') \mid \langle \zeta_2', p_2' \rangle \in p_2(\zeta) \}) \\
\leq d(p_1, p_2).
\]

From Michael’s theorem (Theorem A.28(2)) we can conclude that

\[
d\left( \bigcup \{ \zeta_1' \theta(p_1')(\zeta_1') \mid \langle \zeta_1', p_1' \rangle \in p_1(\zeta) \}, \bigcup \{ \zeta_2' \theta(p_2')(\zeta_2') \mid \langle \zeta_2', p_2' \rangle \in p_2(\zeta) \} \right) \\
\leq d(p_1, p_2). \quad \Box
\]

The unique fixed point of this contractive function from a complete metric space to itself is the linearise operator.

**Definition 75.** The function \( \text{lin} : \mathcal{P} \rightarrow \Sigma \rightarrow \mathcal{P}_\infty(\Sigma^\infty) \) is defined by

\[
\text{lin} = \text{fix}(\Theta).
\]

According to Banach’s theorem, the linearise operator \( \text{lin} \) is the unique element of the complete metric space \( \mathcal{P} \rightarrow \Sigma \rightarrow \mathcal{P}_\infty(\Sigma^\infty) \) satisfying

\[
\text{lin}(p)(\zeta) = \begin{cases} \{ \zeta \} & \text{if } p = \sqrt{\cdot}, \\ \bigcup \{ \zeta' \theta \text{lin}(p')(\zeta') \mid \langle \zeta', p' \rangle \in p(\zeta) \} & \text{otherwise}. \end{cases}
\]

Having introduced the linearise operator we have left to link the operational and intermediate semantic models. We will prove their relationship by uniqueness of fixed point. We will exploit some of the general theory developed in Appendix B. According to Theorem B.31, a finitely branching labelled transition system, like the one introduced
in Definition 44, induces a compactness preserving semantics transformation (see Definition B.26 and B.29). For the labelled transition system at hand this is the function $\mathcal{T} : [\text{Stat} \times \Sigma \rightarrow \mathcal{P}(\Sigma^\infty)]$ defined by

$$
\mathcal{T}(\mathcal{F})([\bar{s}, \zeta]) = \begin{cases} 
\{e\} & \text{if } \bar{s} = e, \\
\bigcup \{\zeta' \mathcal{F}([\bar{s}', \zeta']) \mid [\bar{s}, \zeta] \xrightarrow{\zeta'} [\bar{s}', \zeta']\} & \text{otherwise}.
\end{cases}
$$

According to Theorem B.34, the operational semantics $\mathcal{O}$ given in Section 4.2 is the unique fixed point of the semantics transformation $\mathcal{T}$. Next we show that

$$
\lambda[\bar{s}', \zeta']. \text{lin}(\mathcal{F}(\bar{s}'))(\zeta')
$$

is also a fixed point of $\mathcal{T}$.

**Theorem 76.** For all $\bar{s} \in \text{Stat}_e$ and $\zeta \in \Sigma$,

$$
\mathcal{T}(\lambda[\bar{s}', \zeta']. \text{lin}(\mathcal{F}(\bar{s}'))(\zeta'))([\bar{s}, \zeta]) = \text{lin}(\mathcal{F}(\bar{s}))(\zeta).
$$

**Proof.** We distinguish two cases.

- If $\bar{s} = e$ then

$$
\mathcal{T}(\lambda[\bar{s}', \zeta']. \text{lin}(\mathcal{F}(\bar{s}'))(\zeta'))([e, \zeta])
= \{e\}
= \text{lin}(\sqrt{e})(\zeta)
= \text{lin}(\mathcal{F}(e))(\zeta).
$$

- If $\bar{s} \neq e$ then

$$
\mathcal{T}(\lambda[\bar{s}', \zeta']. \text{lin}(\mathcal{F}(\bar{s}'))(\zeta'))([\bar{s}, \zeta])
= \bigcup \{\zeta' \text{lin}(\mathcal{F}(\bar{s}'))(\zeta') \mid [\bar{s}, \zeta] \xrightarrow{\zeta'} [\bar{s}', \zeta']\}
= \text{lin}(\lambda\zeta, \lambda[\zeta', \mathcal{F}(\bar{s}')]([\bar{s}, \zeta] \xrightarrow{\zeta'} [\bar{s}', \zeta']))(\zeta)
= \text{lin}(\mathcal{F}(\bar{s}))(\zeta).
$$

By uniqueness of fixed point we have the following

**Corollary 77.** For all $\bar{s} \in \text{Stat}_e$ and $\zeta \in \Sigma$, $\mathcal{O}([\bar{s}, \zeta]) = \text{lin}(\mathcal{F}(\bar{s}))(\zeta)$.

**Proof.** Immediate consequence of Theorem B.34 and 76. \qed

Combining the above results we arrive at

**Theorem 78.** $\mathcal{O} = \text{lin} \circ \mathcal{D}$.
Proof. For all \( s \in \text{Stat} \) and \( \xi \in \Sigma \),
\[
\begin{align*}
\mathcal{C}(s)(\xi) \\
= \mathcal{C}([s, \xi]) \\
= \text{lin}(\mathcal{I}(s))(\xi) \quad \text{[Corollary 77]} \\
= \text{lin}(\mathcal{D}(s))(\xi) \quad \text{[Corollary 71]}. \\
\end{align*}
\]

5. Unbounded nondeterminism and nontermination

We investigate a timed specification language by enriching the language studied in Section 3 with two basic constructions which involve time and by studying its semantics. We add the wait statement and the dense choice. As we saw in the previous section, the addition of the parallel composition to the language of Section 3 introduced bounded nondeterminism. The labelled transition system defining the operational semantics was shown to be finitely branching. The dense choice gives rise to unbounded nondeterminism: there are configurations with uncountably many outgoing transitions in the system inducing the operational semantics of the timed language. As we will see, the presence of unbounded nondeterminism complicates the development of the denotational semantics and the proof that the operational and the denotational models coincide.

The wait statement \texttt{wait} \texttt{e} is a basic construction. Its execution amounts to evaluating the expression \texttt{e}, which results in a nonnegative real number \( r \), and waiting for \( r \) seconds. We stipulate that the execution of the assignment statement and the skip statement and the evaluation of an expression and a Boolean expression takes no time. Consequently, the execution of the statement \texttt{skip ; \texttt{wait} 2.16} takes only 2.16 s. The other timed construct added to the language is the dense choice. This highly nondeterministic construct may give rise to a choice between an uncountably infinite number of alternatives. The execution of the dense choice \texttt{t in } [e_1, e_2] \texttt{; \texttt{skip} ; \texttt{wait} t} takes between 2 and 3 s. A minor variation on this construct has been studied by Baeten and Bergstra in [7].

As usual, we define the operational semantics by means of a labelled transition system. As before, the configurations of the system are pairs of statements and states. Here, a state not only assigns to each variable its value, a natural number, but it also associates to each time variable its value, a nonnegative real number. A label of the system is either a state, as before, or a nonnegative real number. The latter is used to model the passage of time. For both timed constructs we add an axiom. As we will see, the dense choice causes the labelled transition system to be infinitely branching.
Like the operational semantics, the denotational semantics assigns to each statement of the timed language a function mapping states to sets of action sequences – an action being a state or a nonnegative real number. As we already observed in Section 4, we have to restrict ourselves to compact sets. If we were to endow the action sequences with the Baire metric, then the denotational semantics of a dense choice would in general not deliver a compact set. Therefore, we provide the action sequences with a more refined metric. The set of action sequences and its metric are defined as the unique solution of a recursive equation. One of the key ingredients of this equation is the set of nonnegative real numbers endowed with the Euclidean metric.

Since the labelled transition system defining the operational semantics is not finitely branching, we cannot exploit the theory developed in Appendix B to link the operational and denotational semantics. We add some structure to the labelled transition system by endowing both the configurations and the actions with a complete metric. The obtained enriched system we call a metric labelled transition system. We show that this enhanced system is compactly branching, that is, every configuration has a compact set of outgoing transitions, and (that transitioning is) nonexpansive. Now we can exploit the theory developed in Appendix C to relate the operational and denotational semantics. The theory presented in that appendix is a generalisation of the one of Appendix B. It is not restricted to finitely branching labelled transition systems but applies to compactly branching and nonexpansive metric labelled transition systems (a finitely branching system of which the configurations and actions are endowed with the discrete metric is a compactly branching and nonexpansive metric system).

As usual, we first introduce the language in Section 5.1. Next, we give an operational and a denotational semantics in Section 5.2 and 5.3. Finally, we prove the two to coincide in Section 5.4.

5.1. Language definition

The language we study is obtained from the one of Section 3 by adding the wait statement and the dense choice construct.

For the introduction of these timed constructs we presuppose a set \((t \in)\) \(TVar\) of time variables. These time variables will take a nonnegative real number as their values. We denote the set of nonnegative real numbers by \((r \in)\) \(\mathbb{R}_+\).

**Definition 79.** The set \((s \in)\) \(Stat\) of statements is defined by

\[
s ::= v := e \mid \text{skip} \mid t \in [e,e] \mid \text{wait } e \mid s \mid s; s \mid \text{if } b \text{ then } s \text{ else } s \text{ fi} \mid \text{while } b \text{ do } s \text{ od}
\]

To simplify matters a little, we restrict the language in the following ways.
- The expression \(e\) in the assignment statement \(v := e\) only contains ordinary variables and no time variables. If we were to consider also expressions with time variables, then we had to restrict ourselves to nonexpansive (with respect to the Euclidean metric) expressions in the denotational semantics.
• Similarly, the Boolean expression \( b \) in the if statement \( \text{if } b \text{ then } s \text{ else } s \text{ fi} \) and in the while statement \( \text{while } b \text{ do } s \text{ od} \) contains no time variables.

• We only consider wait statements of the form \( \text{wait } t \). Note that the expression \( t \) is nonexpansive.

• We focus on dense choices of the form \( t \in [r_1, r_2] \) with \( r_1 \leq r_2 \).

We conclude this section with an example.

\textbf{Example 80.} In this example we specify three clocks. The statement
\[
\text{while true do wait } 1; v := v + 1 \text{ od}
\]
describes a clock which progresses with absolute precision. The statement
\[
\text{while true do } t \in [0.98, 1.00]; \text{wait } t; v := v + 1; \text{wait } 1 - t \text{ od}
\]
specifies a clock with some fluctuation. A clock accumulating the errors is described by the statement
\[
\text{while true do } t \in [0.98, 1.00]; \text{wait } t; v := v + 1 \text{ od}.
\]

\textbf{5.2. Operational semantics}

As usual, the operational semantics is defined by means of a labelled transition system. The system is nondeterministic and nonterminating. As we will see, it is not finitely branching.

Like in the previous sections, a configuration of the labelled transition system is a statement and a state. Here, a state assigns to each variable a natural number and to each time variable a nonnegative real number.

\textbf{Definition 81.} The set \((\zeta, \tau) \in \Sigma\) of states is defined by
\[
\Sigma = (\text{Var} \rightarrow \mathbb{N}) \times (\text{TVar} \rightarrow \mathbb{R}_+).
\]

An action \( \alpha \) of the labelled transition system is either a state (like in the foregoing sections) or a nonnegative real number. By means of a nonnegative real number we model the passage of time. For example, if \( \tau(t) = 1 \) then we have the transition
\[
[\text{wait } t, \zeta, \tau] \xrightarrow{1} [e, \zeta, \tau].
\]
The transition relation of the labelled transition system is presented in the following definition.

\textbf{Definition 82.} The transition relation \( \rightarrow \) is defined by the following axioms and rules.

\begin{enumerate}
\item \( [v := e, \zeta, \tau] \xrightarrow{\text{rule } 1} \big[e, \zeta[n/e], \tau\big] \) where \( n = \delta(e)(\zeta) \),
\item \( [\text{skip}, \zeta, \tau] \xrightarrow{\text{rule } 2} [e, \zeta, \tau], \)
\end{enumerate}
In (3) we describe the dense choice construct. Consider:

Exercise 83. Show that the labelled transition system is not finitely branching.

Some comments:
- The axioms (1) and (2) have already seen several times.
- In (3) we describe the dense choice construct. Consider \( t \) in \([n_1,r_2]\) in the current state \( \langle \varsigma, \tau \rangle \). By assumption, \( n_1 \leq r_2 \). The execution of this dense choice amounts to nondeterministically choosing a value \( r \) from the nonempty interval \([n_1,r_2]\). Next, we update the state \( \langle \varsigma, \tau \rangle \) by assigning to the time variable \( t \) the value \( r \), resulting in the new state \( \langle \varsigma, \tau \{ r/t \} \rangle \).
- The execution of the statement \( \text{wait} \) terminates successfully after \( \tau(t) \) seconds, where \( \tau(t) \) is the value of the time variable \( t \) in the current state \( \langle \varsigma, \tau \rangle \). This is expressed by axiom (4). Note that the state does not change.
- The rules of (5) are similar to the ones of the previous sections. Note that the nonnegative real numbers – modelling the passage of time – are dealt with in the same way as the states.
- The rules of (6) are the obvious modifications of the rules for the if statement we have seen before.
- The rules and the axiom of (7) are also straightforward modifications.

Clearly, the above-introduced labelled transition system is nondeterministic and non-terminating.

Exercise 83. Show that the labelled transition system is not finitely branching.
given by
\[
C([S, \tau]) = \{ x_1 x_2 \cdots x_n \mid [S, \tau] = [S_0, \tau_0] \xrightarrow{x_1} [S_1, \tau_1] \xrightarrow{x_2} \cdots \xrightarrow{x_n} [E, \tau_n] \} \cup \\
\{ x_1 x_2 \cdots \mid [S, \tau] = [S_0, \tau_0] \xrightarrow{x_1} [S_1, \tau_1] \xrightarrow{x_2} \cdots \}.
\]

**Definition 84.** The function \( C : Stat \rightarrow \Sigma \rightarrow \mathcal{P}_n((\Sigma \cup \mathbb{R}_+)^\infty) \) is defined by
\[
C(s) = \lambda(x, \tau).C([x, \tau]).
\]

### 5.3. Denotational semantics

Like in the operational semantics, also in the denotational semantics we assign to each statement a function mapping every state to a set of action sequences, where an action is either a state or a nonnegative real number. To obtain a complete metric space, we only consider compact sets of action sequences. As we will see below, we need a more refined metric on the action sequences than the Baire metric (see Exercise 98 and 99) to make sure that the denotational semantics delivers compact sets. This metric is introduced in

**Definition 85.** The complete metric space \((\sigma \in \mathcal{A}^\infty)\) is defined as the solution of the recursive equation
\[
\mathcal{A}^\infty \cong \{ \epsilon \} + (\mathcal{A} \times \frac{1}{2} \cdot \mathcal{A}^\infty),
\]
where the complete metric space \((\omega \in \mathcal{A})\) of actions is defined by
\[
\mathcal{A} = \Sigma + \mathbb{R}_+.
\]

Recall that the set \(\Sigma\) of states is defined by
\[
\Sigma = (Var \rightarrow \mathbb{N}) \times (TVar \rightarrow \mathbb{R}_+).
\]

We endow the sets \(Var, TVar,\) and \(\mathbb{N}\) with the discrete metric and the set \(\mathbb{R}_+\) with the Euclidean metric (restricted to \(\mathbb{R}_+\)). From Theorem D.3 we can deduce that the above equation has a unique solution.

We start with the modelling of the basic constructs of the language: the assignment statement, the skip statement, the dense choice, and the wait statement. Like the operational semantics, the denotational semantics assigns to the assignment statement \(v := e\) the function
\[
\lambda(x, \tau).\{ (\xi[v/v], \tau) \} \quad \text{where} \; n = E(e)(\xi).
\]

The skip statement \(skip\) is mapped to
\[
\lambda(x, \tau).\{ (\xi, \tau) \}.
\]
The dense choice \( t \) in \([r_1, r_2]\) is modelled by
\[
\lambda(\zeta, \tau).\{\langle \zeta, \tau \{r/t\} \rangle | r \in [r_1, r_2]\}.
\]
Because the set \([r_1, r_2]\) is compact (Proposition A.23), we can deduce that the above set is also compact. To the wait statement \( \text{wait } t \) we associate the function
\[
\lambda(\zeta, \tau).\{t(t)\}.
\]
The if statement can be handled as before. To model the sequential composition denotationally, we introduce a semantic sequential composition
\[
; : \Sigma \times A^\infty \longrightarrow ((\Sigma \rightarrow \mathcal{P}_{nc}(A^\infty)) \rightarrow \mathcal{P}_{nc}(A^\infty))
\]
(see Definition 29). For \( f, g \in \Sigma \longrightarrow \mathcal{P}_{nc}(A^\infty) \), we define the sequential composition of \( f \) and \( g \) by
\[
\lambda(\zeta, \tau).\bigcup\{\sigma;_{(\zeta, t)} g | \sigma \in f(\zeta, \tau)\}.
\]
The fact that this function is nonexpansive and delivers compact sets is discussed later (see Proposition 87 and 89). The semantic sequential composition is defined as the unique fixed point of the function \( \Phi \) introduced in

**Definition 86.** The function \( \Phi : [\Sigma \times A^\infty \longrightarrow ((\Sigma \rightarrow \mathcal{P}_{nc}(A^\infty)) \rightarrow \mathcal{P}_{nc}(A^\infty))] \) is defined by
\[
\Phi(\phi)(\zeta, \tau, \sigma)(f) = \begin{cases} 
  f(\zeta, \tau) & \text{if } \sigma = e, \\
  \langle \zeta', \tau' \rangle \phi(\zeta', \tau', \sigma')(f) & \text{if } \sigma = \langle \zeta', \tau' \rangle \sigma', \\
  r\phi(\zeta, \tau, \sigma')(f) & \text{if } \sigma = r\sigma'.
\end{cases}
\]

To conclude that the function \( \Phi \) is indeed well-defined we have to verify that for all \( \phi \in \Sigma \times A^\infty \longrightarrow ((\Sigma \rightarrow \mathcal{P}_{nc}(A^\infty)) \rightarrow \mathcal{P}_{nc}(A^\infty)) \),
- for all \( \langle \zeta, \tau \rangle \in \Sigma, \sigma \in A^\infty \), and \( f \in \Sigma \longrightarrow \mathcal{P}_{nc}(A^\infty) \), the set \( \Phi(\phi)(\zeta, \tau, \sigma)(f) \) is compact,
- for all \( \langle \zeta, \tau \rangle \in \Sigma \) and \( \sigma \in A^\infty \), the function \( \Phi(\phi)(\zeta, \tau, \sigma) \) is nonexpansive, and
- the function \( \Phi(\phi) \) is nonexpansive.

**Proposition 87.** The set \( \Phi(\phi)(\zeta, \tau, \sigma)(f) \) is compact.

**Proof.** We distinguish three cases.
- If \( \sigma = e \) then \( \Phi(\phi)(\zeta, \tau, \sigma)(f) = f(\zeta, \tau) \), which is by definition a compact set.
- If \( \sigma = \langle \zeta', \tau' \rangle \sigma' \) then \( \Phi(\phi)(\zeta, \tau, \sigma)(f) = \langle \zeta', \tau' \rangle \phi(\zeta', \tau', \sigma')(f) \). Since the set \( \phi(\zeta', \tau', \sigma')(f) \) is compact, also the set \( \langle \zeta', \tau' \rangle \phi(\zeta', \tau', \sigma')(f) \) is compact.
- The case that \( \sigma = r\sigma' \) is similar to the previous one. \( \square \)

**Exercise 88.** Show that the function \( \Phi(\phi)(\zeta, \tau, \sigma) \) is nonexpansive.
Proposition 89. The function $\Phi(\phi)$ is nonexpansive.

Proof. Let $(\xi_1, \tau_1), (\xi_2, \tau_2) \in \Sigma$ and $\sigma_1, \sigma_2 \in A^\infty$. It suffices to verify that for all $f \in \Sigma \rightarrow P_\infty(A^\infty)$, we have that
\[
d(\Phi(\phi)(\xi_1, \tau_1, \sigma_1)(f), \Phi(\phi)(\xi_2, \tau_2, \sigma_2)(f)) \leq d((\xi_1, \tau_1, \sigma_1), (\xi_2, \tau_2, \sigma_2)).
\]
We distinguish the following cases.
- If $\sigma_1 = \epsilon$ and $\sigma_2 = \epsilon$ then
\[
d(\Phi(\phi)(\xi_1, \tau_1, \epsilon)(f), \Phi(\phi)(\xi_2, \tau_2, \epsilon)(f))
= d(f(\xi_1, \tau_1), f(\xi_2, \tau_2))
\leq d((\xi_1, \tau_1), (\xi_2, \tau_2)) \quad [f \ is \ nonexpansive]
\leq d((\xi_1, \tau_1, \sigma_1), (\xi_2, \tau_2, \sigma_2)).
\]
- If $\sigma_1 = (\xi_1', \tau_1')\sigma_1'$ and $\sigma_2 = (\xi_2', \tau_2')\sigma_2'$ then
\[
d(\Phi(\phi)(\xi_1', \tau_1', \sigma_1')(f), \Phi(\phi)(\xi_2', \tau_2', \sigma_2')(f))
= d((\xi_1', \tau_1')\phi(\xi_1, \tau_1, \sigma_1'(f)), (\xi_2', \tau_2')\phi(\xi_2, \tau_2, \sigma_2')(f))
= \max\{d((\xi_1', \tau_1'), (\xi_2', \tau_2')), \frac{1}{2} \cdot d(\phi(\xi_1', \tau_1, \sigma_1'(f)), \phi(\xi_2', \tau_2, \sigma_2')(f))\}
\leq \max\{d((\xi_1', \tau_1', \sigma_1'), (\xi_2', \tau_2', \sigma_2')), \frac{1}{2} \cdot d((\xi_1', \tau_1', \sigma_1'), (\xi_2', \tau_2', \sigma_2'))\}
\quad [\phi \ is \ nonexpansive]
= d(\sigma_1, \sigma_2)
\leq d((\xi_1, \tau_1, \sigma_1), (\xi_2, \tau_2, \sigma_2)).
\]
- If $\sigma_1 = r_1\sigma_1'$ and $\sigma_2 = r_2\sigma_2'$ then
\[
d(\Phi(\phi)(\xi_1, \tau_1, r_1\sigma_1')(f), \Phi(\phi)(\xi_2, \tau_2, r_2\sigma_2')(f))
= d(r_1\phi(\xi_1, \tau_1, \sigma_1')(f), r_2\phi(\xi_2, \tau_2, \sigma_2')(f))
= \max\{d(r_1, r_2), \frac{1}{2} \cdot d(\phi(\xi_1, \tau_1, \sigma_1')(f), \phi(\xi_2, \tau_2, \sigma_2')(f))\}
\leq \max\{d(r_1, r_2), \frac{1}{2} \cdot d((\xi_1, \tau_1, \sigma_1'), (\xi_2, \tau_2, \sigma_2'))\} \quad [\phi \ is \ nonexpansive]
\leq d((\xi_1, \tau_1, \sigma_1), (\xi_2, \tau_2, \sigma_2)). \quad \square
\]
To conclude from Banach’s theorem that $\Phi$ has a unique fixed point we have left to show that the function is a contraction.

Proposition 90. The function $\Phi$ is contractive.

Proof. As usual. \quad \square
Now we can give

**Definition 91.** The function \( ; : \Sigma \times \mathcal{A}^\infty \rightarrow ((\Sigma \rightarrow \mathcal{P}_{\text{nc}}(\mathcal{A}^\infty)) \rightarrow \mathcal{P}_{\text{nc}}(\mathcal{A}^\infty)) \) is defined by

\[
; = \text{fix}(\Phi).
\]

The semantic sequential composition has the usual property.

**Proposition 92.** For all \( f_1, f_2 \in \Sigma \rightarrow \mathcal{P}_{\text{nc}}(\mathcal{A}^\infty) \), \( \sigma \in \mathcal{A}^\infty \), and \( \langle \varsigma, \tau \rangle \in \Sigma \),

\[
d(\sigma ; (\varsigma, \tau) f_1, \sigma ; (\varsigma, \tau) f_2) \leq \begin{cases} 
d(f_1, f_2) & \text{if } \sigma = \varepsilon, \\
\frac{1}{2} \cdot d(f_1, f_2) & \text{otherwise}.
\end{cases}
\]

**Proof.** Similar to the proof of Proposition 32. \( \square \)

Next, we give the semantic counterpart of the while statement. As before, it is defined as the unique fixed point of the function defined in

**Definition 93.** Let \( \beta \in \Sigma \rightarrow \{\text{true}, \text{false}\} \) and \( f \in \Sigma \rightarrow \mathcal{P}_{\text{nc}}(\mathcal{A}^\infty) \) such that \( \varepsilon \notin f(\varsigma, \tau) \) for all \( \langle \varsigma, \tau \rangle \in \Sigma \). The function \( \Psi(\beta, f) : [\Sigma \rightarrow \mathcal{P}_{\text{nc}}(\mathcal{A}^\infty)] \) is defined by

\[
\Psi(\beta, f)(\psi)(\varsigma, \tau) = \begin{cases} 
\bigcup \{\sigma ; (\varsigma, \tau) \psi | \sigma \in f(\varsigma, \tau)\} & \text{if } \beta(\varsigma) = \text{true}, \\
\{\langle \varsigma, \tau \rangle\} & \text{if } \beta(\varsigma) = \text{false}.
\end{cases}
\]

To deduce that this function \( \Psi(\beta, f) \) is well-defined we have to check that for all \( \psi \in \Sigma \rightarrow \mathcal{P}_{\text{nc}}(\mathcal{A}^\infty) \),

- for all \( \langle \varsigma, \tau \rangle \in \Sigma \), the set \( \Psi(\beta, f)(\psi)(\varsigma, \tau) \) is compact, and
- the function \( \Psi(\beta, f)(\psi) \) is nonexpansive.

**Proposition 94.** The set \( \Psi(\beta, f)(\psi)(\varsigma, \tau) \) is compact.

**Proof.** Obviously, the set \( \{\langle \varsigma, \tau \rangle\} \) is compact.

For all \( \sigma \in f(\varsigma, \tau) \), the set \( \sigma ; (\varsigma, \tau) \psi \) is compact. Since the set \( f(\varsigma, \tau) \) is compact and the function \( \lambda \sigma . \sigma ; (\varsigma, \tau) \psi \) is nonexpansive, we can conclude from Alexandroff’s theorem that \( \{\sigma ; (\varsigma, \tau) \psi | \sigma \in f(\varsigma, \tau)\} \) is also a compact set. By Michael’s theorem, the set \( \bigcup \{\sigma ; (\varsigma, \tau) \psi | \sigma \in f(\varsigma, \tau)\} \) is compact. \( \square \)

**Proposition 95.** The function \( \Psi(\beta, f)(\psi) \) is nonexpansive.

**Proof.** We prove for all \( \langle \varsigma, \tau_1 \rangle, \langle \varsigma, \tau_2 \rangle \in \Sigma \) that

\[
d(\Psi(\beta, f)(\psi)(\varsigma, \tau_1), \Psi(\beta, f)(\psi)(\varsigma, \tau_2)) \leq d(\tau_1, \tau_2).
\]
We distinguish the following two cases.

- Assume $\beta(\zeta) = \text{true}$. Then,

$$d(\Psi(\beta, f)(\psi)(\zeta, \tau_1), \Psi(\beta, f)(\psi)(\zeta, \tau_2))$$

$$= d(\bigcup \{ \sigma_1 :_{(\zeta, \tau_1)} \psi \mid \sigma_1 \in f(\zeta, \tau_1) \}, \bigcup \{ \sigma_2 :_{(\zeta, \tau_2)} \psi \mid \sigma_2 \in f(\zeta, \tau_2) \})$$

$$\leq d(\{ \sigma_1 :_{(\zeta, \tau_1)} \psi \mid \sigma_1 \in f(\zeta, \tau_1) \}, \{ \sigma_2 :_{(\zeta, \tau_2)} \psi \mid \sigma_2 \in f(\zeta, \tau_2) \})$$

[Michael’s theorem].

Let $\sigma_1 \in f(\zeta, \tau_1)$. Then there exists a $\sigma_2 \in f(\zeta, \tau_2)$ such that

$$d(\sigma_1, \sigma_2)$$

$$\leq d(f(\zeta, \tau_1), f(\zeta, \tau_2))$$

$$\leq d(\tau_1, \tau_2) \ [f \text{ is nonexpansive}].$$

Consequently,

$$d(\sigma_1 :_{(\zeta, \tau_1)} \psi \sigma_2 :_{(\zeta, \tau_2)} \psi)$$

$$\leq d(\langle \sigma_1, \zeta, \tau_1 \rangle, \langle \sigma_2, \zeta, \tau_2 \rangle) \ [; \text{ is nonexpansive}]$$

$$\leq d(\tau_1, \tau_2) \ [\text{see above}].$$

- Let $\beta(\zeta) = \text{false}$. In this case,

$$d(\Psi(\beta, f)(\psi)(\zeta, \tau_1), \Psi(\beta, f)(\psi)(\zeta, \tau_2))$$

$$= d(\{ \langle \zeta, \tau_1 \rangle \}, \{ \langle \zeta, \tau_2 \rangle \})$$

$$= d(\tau_1, \tau_2). \ \Box$$

To conclude that the function $\Psi(\beta, f)$ has a unique fixed point, we still need to show that it is a contraction.

**Proposition 96.** The function $\Psi(\beta, f)$ is contractive.

**Proof.** Similar to the proof of Proposition 35, exploiting Proposition 92. $\Box$

Having introduced the semantic operators, we are ready to give the denotational semantics.

**Definition 97.** The function $\mathcal{D} : \text{Stat} \to \Sigma \longrightarrow \mathcal{P}_{\text{he}}(A^\infty)$ is defined by

$$\mathcal{D}(v := e) = \lambda(\zeta, \tau).\{ \langle \zeta \{ n/v \}, \tau \} \} \ \text{where} \ n = \mathcal{E}(e)(\zeta),$$

$$\mathcal{D}(\text{skip}) = \lambda(\zeta, \tau).\{ \langle \zeta, \tau \} \}.$$
\[ D(t \text{ in } [r_1, r_2]) = \lambda \langle \zeta, \tau \rangle. \{ \langle \zeta, \tau \{ r/t \} \rangle | r \in [r_1, r_2] \}, \]
\[ D(\text{wait } t) = \lambda \langle \zeta, \tau \rangle. \{ \tau(t) \}, \]
\[ D(s_1; s_2) = \lambda \langle \zeta, \tau \rangle. \bigcup \{ \sigma \in D(s_1)(\zeta, \tau) \}, \]
\[ D(\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ if}) = \lambda \langle \zeta, \tau \rangle. \begin{cases} D(s_1)(\zeta, \tau) & \text{if } A(b)(\zeta) = \text{true}, \\ D(s_2)(\zeta, \tau) & \text{if } A(b)(\zeta) = \text{false}, \end{cases} \]
\[ D(\text{while } b \text{ do } s \text{ od}) = \text{fix}(\Psi(A(b), D(s))). \]

**Exercise 98.** Prove that for all \( s \in \text{Stat} \) and \( \langle \zeta, \tau \rangle \in \Sigma \), the set \( D(s)(\zeta, \tau) \) is compact and the function \( D(s) \) is nonexpansive.

**Exercise 99.** Show that if we were to endow the set \((\Sigma \cup \mathbb{R}_+)^\infty\) with the Baire metric instead, then the set \( D(s)(\zeta) \) would in general not be compact.

### 5.4. Relating \( \mathcal{O} \) and \( D \)

Since the labelled transition system is not finitely branching (Exercise 83), we cannot exploit the theory developed in Appendix B to link the operational and denotational semantics. However, by endowing the configurations and the actions with suitable complete metrics, we obtain a metric labelled transition system which is compactly branching and nonexpansive as we will see below. Then we can use the theory of Appendix C to relate the operational and denotational model.

We start with defining a metric for the configurations. A configuration consists of statement and a state. The set \( \text{Stat} \) is endowed with the discrete metric and the set \( \Sigma \) with the metric introduced in Section 5.3 (obtained by endowing the sets \( \text{Var}, \text{TVar}, \) and \( \mathbb{N} \) with the discrete metric and \( \mathbb{R}_+ \) with the Euclidean one).

Next, we turn the set of actions into a complete metric space. The states are endowed with the metric mentioned above and the nonnegative real numbers with the Euclidean metric (see Definition 85).

To prove that the metric system is compactly branching and nonexpansive, we extend the complexity function of Definition 40 as follows.

**Definition 100.** The function \( \text{comp} : \text{Stat} \rightarrow \mathbb{N} \) is defined by

\[ \text{comp}(e) = 0, \]
\[ \text{comp}(v := e) = 1, \]
\[ \text{comp}(\text{skip}) = 1, \]
\[ \text{comp}(t \text{ in } [r_1, r_2]) = 1, \]
\[ \text{comp}(\text{wait } t) = 1, \]
\[ \text{comp}(s_1; s_2) = \text{comp}(s_1) + 1, \]
\[ \text{comp}(\text{if } b \text{ then } s_1 \text{ else } s_2 \text{ if}) = \max\{\text{comp}(s_1), \text{comp}(s_2)\} + 1, \]
\[ \text{comp}(\text{while } b \text{ do } s \text{ od}) = \text{comp}(s) + 3. \]
Now we are ready to prove the main result of this section.

**Theorem 101.** The metric labelled transition system is compactly branching and non-expansive.

**Proof.** We prove that for all $\bar{s} \in \text{Stat}_k$,
- for all $\langle \bar{s}, \tau \rangle \in \Sigma$, the set $\mathcal{S}(\langle \bar{s}, \tau \rangle)$ is compact, and
- for all $\langle \bar{s}, \tau_1 \rangle, \langle \bar{s}, \tau_2 \rangle \in \Sigma$, $d(\mathcal{S}(\langle \bar{s}, \tau_1 \rangle), \mathcal{S}(\langle \bar{s}, \tau_2 \rangle)) \leq d(\tau_1, \tau_2)$
by induction on $\text{comp}(\bar{s})$. We distinguish the following cases.
- For $\bar{s} = \text{E}$ the above is vacuously true.
- Let $\bar{s} = v := e$. Clearly the set $\mathcal{S}(\langle v := e, \tau \rangle) = \{\langle \text{E, } \tau \rangle, \langle e, \tau \rangle\}$, where $n = \delta(e)(\bar{s})$, is compact. Furthermore,
  $$d(\mathcal{S}(\langle v := e, \tau \rangle), \mathcal{S}(\langle v := e, \tau \rangle)) = d(\mathcal{S}(\langle v := e, \tau \rangle), \mathcal{S}(\langle v := e, \tau \rangle)) = \frac{1}{2} \cdot d([\text{E, } \tau], [e, \tau])$$
- The case that $\bar{s} = \text{skip}$ is similar to the previous case.
- Suppose $\bar{s} = t$ in $[r_1, r_2]$. Then
  $$\mathcal{S}(\langle t \text{ in } [r_1, r_2], \tau \rangle) = \{\langle \text{E, } \tau \rangle, [t, \tau] \}$$
Because for all $r_1, r_2 \in \mathbb{R}_+$,
  $$d(\mathcal{S}(\langle t \text{ in } [r_1, r_2], \tau \rangle), \mathcal{S}(\langle t \text{ in } [r_1, r_2], \tau \rangle)) = d(r_1, r_2),$$
the function $\lambda r. \langle \text{E, } \tau \rangle, [t, \tau] \rangle$ is nonexpansive. Since the set $[r_1, r_2]$ is compact (Proposition A.23), we can conclude from Alexandroff’s theorem that the set $\mathcal{S}(\langle t \text{ in } [r_1, r_2], \tau \rangle)$ is also compact.
Furthermore, for all $r \in [r_1, r_2]$, we have that
  $$d(\mathcal{S}(\langle t \rangle, \tau_1 \rangle), \mathcal{S}(\langle t \rangle, \tau_2 \rangle)) = d(\tau_1, \tau_2).$$
- Let $\bar{s} = \text{wait } t$. Clearly, the set
  $$\mathcal{S}(\langle \text{wait } t, \tau \rangle) = \{\langle t, \text{E, } \tau \rangle\}$$
is compact. Furthermore,
\[
d(\mathcal{S}(\text{wait } t, \varnothing, \tau_1), \mathcal{S}(\text{wait } t, \varnothing, \tau_2)) = d(\{(\tau_1(t), [E, \varnothing, \tau_1])\}, \{(\tau_2(t), [E, \varnothing, \tau_2])\})
\]
\[
= \max\{d(\tau_1(t), \tau_2(t)), \frac{1}{2} \cdot d([E, \varnothing, \tau_1], [E, \varnothing, \tau_2])\}
\]
\[
\leq d(\tau_1, \tau_2).
\]
- If \(\bar{s} = s_1; s_2\) then
\[
\mathcal{S}([s_1; s_2, \varnothing, \tau])
\]
\[
= \{(x, [s_2', \varnothing', \tau']) \mid (x, [E, \varnothing', \tau']) \in \mathcal{S}([s_1, \varnothing, \tau])\} \cup
\]
\[
\{(x, [s_1', s_2', \varnothing', \tau']) \mid (x, [s_1', \varnothing', \tau']) \in \mathcal{S}([s_1, \varnothing, \tau])\}.
\]
By induction, the set \(\mathcal{S}([s_1, \varnothing, \tau])\) is compact. Since the statement part of a configuration is endowed with the discrete metric, we can conclude that the above set is also compact.

Let \((x_1, [s_2, \varnothing', \tau_1']) \in \mathcal{S}([s_1; s_2, \varnothing, \tau_1])\). Then \((x_1, [E, \varnothing', \tau_1']) \in \mathcal{S}([s_1, \varnothing, \tau_1])\). Hence, there exists a \((x_2, [E, \varnothing_2', \tau_2']) \in \mathcal{S}([s_1, \varnothing, \tau_2])\) such that
\[
d((x_1, [E, \varnothing', \tau_1']), (x_2, [E, \varnothing_2', \tau_2']))
\]
\[
\leq d(\mathcal{S}([s_1, \varnothing, \tau_1]), \mathcal{S}([s_1, \varnothing, \tau_2]))
\]
\[
\leq d(\tau_1, \tau_2) \quad \text{[induction].}
\]
Consequently, \((x_2, [s_2, \varnothing_2', \tau_2']) \in \mathcal{S}([s_1; s_2, \varnothing, \tau_2])\) and
\[
d((x_1, [s_2, \varnothing', \tau_1']), (x_2, [s_2, \varnothing_2', \tau_2'])) \leq d(\tau_1, \tau_2).
\]
The other case can be dealt with similarly.
- The cases \(\bar{s} = \text{if } b \text{ then } s_1 \text{ else } s_2 \text{ fi}\) and \(\bar{s} = \text{while } b \text{ do } s \text{ od}\) follow by induction.

Having proved that the metric system is compactly branching and nonexpansive, we can exploit the results of Appendix C to prove the operational and denotational semantics to be equal.

We extend the denotational semantics as follows.

**Definition 102.** The function \(\mathcal{D}: \text{Stat}_e \times \Sigma \rightarrow \mathcal{P}_{nc}(A^\infty)\) is defined by

\[
\mathcal{D}([E, \varnothing, \tau]) = \{\varepsilon\},
\]
\[
\mathcal{D}([s, \varnothing, \tau]) = \mathcal{D}(s)(\varnothing, \tau).
\]
According to Theorem C.30, the compactly branching and nonexpansive metric labelled transition system induces a compactness and nonexpansiveness preserving semantics transformation $\mathcal{T} : [\text{Stat}_k \times \Sigma \to \mathcal{P}_{\text{nc}}(A^\infty)]$ defined by

$$\mathcal{T}(\mathcal{D})([\tilde{s}, \zeta, \tau]) = \begin{cases} \{e\} & \text{if } [\tilde{s}, \zeta, \tau] \not\rightarrow, \\ \bigcup \{ \mathcal{D}([\tilde{s}', \zeta', \tau']) | [\tilde{s}, \zeta, \tau] \not\rightarrow [\tilde{s}', \zeta', \tau'] \} & \text{otherwise}. \end{cases}$$

By Theorem C.33, the operational semantics is the unique fixed point of $\mathcal{T}$. Hence, we have left to show that also the denotational semantics is a fixed point of the semantics transformation.

**Theorem 103.** $\mathcal{T}(\mathcal{D}) = \mathcal{D}$.

**Proof.** We prove that for all $\tilde{s} \in \text{Stat}_k$ and $\langle \zeta, \tau \rangle \in \Sigma$,

$$\mathcal{T}(\mathcal{D})([\tilde{s}, \zeta, \tau]) = \mathcal{D}([\tilde{s}, \zeta, \tau])$$

by induction on $\text{comp}(\tilde{s})$. We only consider the following case.

$$\mathcal{T}(\mathcal{D})([t \in [r_1, r_2], \zeta, \tau])$$

$$= \bigcup \{ \langle \zeta, \tau \{r/t\} \rangle \mathcal{D}(\langle e, \zeta, \tau \{r/t\} \rangle) | r \in [r_1, r_2] \}$$

$$= \{ \langle \zeta, \tau \{r/t\} \rangle | r \in [r_1, r_2] \}$$

$$= \mathcal{D}([t \in [r_1, r_2], \zeta, \tau]). \quad \Box$$

Putting the pieces together we arrive at

**Corollary 104.** $\mathcal{O} = \mathcal{D}.$

**Proof.** For all $s \in \text{Stat}$ and $\langle \zeta, \tau \rangle \in \Sigma$,

$$\mathcal{O}(s)(\zeta, \tau)$$

$$= \mathcal{O}([s, \zeta, \tau])$$

$$= \mathcal{D}([s, \zeta, \tau]) \quad \text{[Theorem 103 and C.33]}$$

$$= \mathcal{D}(s)(\zeta, \tau). \quad \Box$$

**Appendix A. Metric spaces**

As we pointed out in the introduction of Section 3, to model recursive constructs like the while statement in a compositional way, we exploit mathematical structures with fixed points. In this paper, we use metric spaces for that purpose. A metric space
consists of a nonempty set and a metric: a function assigning to each pair of elements of the set their distance and satisfying certain natural conditions. For the existence of fixed points we rely on Banach’s fixed point theorem [18]. This theorem roughly tells us the following. Let $X$ be a metric space. Suppose $f$ is a function from $X$ to $X$. This we denote by $f: [X]$ from now on. Assume furthermore that the function $f$ is contractive. Then $f$ has at most one fixed point. If the space $X$ is complete, then $f$ has exactly one fixed point. Metric spaces and Banach’s theorem were already used to model recursive program schemes by Arnold and Nivat (see, for example, [6,43]).

Not only do we exploit Banach’s theorem to model recursive constructs. We also use the theorem to define operators on (finite and) infinite objects. These infinite entities, like (countably) infinite sequences, we use to model nonterminating computations. For example, the sequential composition of finite or infinite sequences over a set $A$

$$\varepsilon': \mathbb{A}^\infty \times \mathbb{A}^\infty \rightarrow \mathbb{A}^\infty$$

satisfying

$$\sigma_1; \sigma_2 = \begin{cases} \sigma_2 & \text{if } \sigma_1 = \varepsilon, \\ a(\sigma_1'; \sigma_2) & \text{if } \sigma_1 = a\sigma_1', \end{cases}$$

can be defined by means of Banach’s theorem. Note that the above is not a definition by induction on the length of $\sigma_1$, since $\sigma_1$ might be infinite. We introduce a function

$$\Phi: [\mathbb{A}^\infty \times \mathbb{A}^\infty \rightarrow \mathbb{A}^\infty]$$

defined by

$$\Phi(\phi)(\sigma_1, \sigma_2) = \begin{cases} \sigma_2 & \text{if } \sigma_1 = \varepsilon, \\ a\phi(\sigma_1', \sigma_2) & \text{if } \sigma_1 = a\sigma_1', \end{cases}$$

As we will see, we can turn the set $\mathbb{A}^\infty \times \mathbb{A}^\infty \rightarrow \mathbb{A}^\infty$ into a complete metric space such that the function $\Phi$ is contractive. According to Banach’s theorem, $\Phi$ has a unique fixed point: the sequential composition ‘;’. This way of defining operators has been used extensively by Kok and Rutten [34].

Furthermore, we exploit the unique fixed point proof principle, a proof principle based on Banach’s theorem, to relate semantic models. This proof principle is generally used to show that two elements of a metric space are equal. Let $X$ be a metric space, not necessarily complete. To show that the elements $x$ and $y$ of $X$ are equal, we introduce a function $f: [X]$, show that $f$ is a contraction, and verify that both $x$ and $y$ are a fixed point of $f$. Since a contractive function from a metric space to itself has at most one fixed point according to Banach’s theorem, we can conclude that $x$ and $y$ must be equal. How this proof principle can be employed to relate operational semantic models defined by means of labelled transition systems to other semantic models is discussed in detail in Appendix B. This proof principle was first exploited systematically to relate semantic models by Kok and Rutten in [34].
In our semantic models, we use the powerset construction to model nondeterminism. In the denotational setting, we exploit the following construction. Given a metric space \( X \), we endow the set \( \mathcal{P}_n(X) \) of nonempty subsets of \( X \) with the Hausdorff metric [30]. As we will see, the empty set can easily be added, but it is technically more convenient to consider only nonempty sets at first. We will present an example showing that \( \mathcal{P}_n(X) \) provided with the Hausdorff metric does not give rise to a metric space in general. However, if we restrict ourselves to the set \( \mathcal{P}_n^c(X) \) of nonempty and compact subsets of the space \( X \), then we do get a metric space. Compactness is a generalisation of finiteness. Every finite set is compact, but the converse is not true. However, for every compact set we can find a finite set which is arbitrary close to it. In this paper, we will often exploit the following two properties of compact sets. Firstly, Michael’s theorem [41] roughly tells us that a compact union of compact sets delivers compact sets and that it is nonexpansive. Secondly, the nonexpansive image of a compact set is again compact – a result due to Alexandroff [1]. The construction of taking the nonempty and compact subsets of a metric space and endowing them with the Hausdorff metric preserves completeness, as was first shown by Kuratowski [37]. This allows us to apply Banach’s theorem also in the presence of spaces of subsets. Nivat already employed the Hausdorff metric in [43]. Compactness was used by De Bakker and Zucker to model fairness in [17].

The rest of this appendix is organised as follows. In Section A.1, we give the definition of a metric space, we present three examples of metrics, and we introduce some operations on metric spaces. In Section A.2, we discuss Banach’s fixed point theorem and its two key ingredients: completeness and contractiveness. In Section A.3, we focus on subsets endowed with the Hausdorff metric. Also compactness is studied in this section. In Section A.4, we concentrate on nonexpansive functions. In order not to burden the presentation we will not present the notions and results in their most general form, but in a way that suits our purposes best. For a proof of Proposition A.23, which employs particular properties of the real numbers, we refer the reader to, for example, [51].

\section{A.1. Metric spaces}

We start with the definition of the most basic notion: a metric space.

\begin{definition}
A \textit{metric space} is a pair \( \langle X,d_X \rangle \) consisting of
\begin{itemize}
\item a nonempty set \( X \) and
\item a function \( d_X:X \times X \to [0,1] \), called \textit{metric}, satisfying
\end{itemize}
\begin{enumerate}
\item for all \( x,y \in X \), \( d_X(x,y)=0 \) if and only if \( x=y \),
\item for all \( x,y \in X \), \( d_X(x,y)=d_X(y,x) \), and
\item for all \( x,y,z \in X \), \( d_X(x,z) \leq d_X(x,y) + d_X(y,z) \).
\end{enumerate}

To simplify notations, we shall usually write \( X \) instead of \( \langle X,d_X \rangle \) and denote the metric of a metric space \( X \) by \( d_X \). Three examples of metrics are presented in
Example A.2.

(1) Let $X$ be a nonempty set. The discrete metric $d_X : X \times X \to [0, 1]$ is defined by

$$d_X(x_1, x_2) = \begin{cases} 0 & \text{if } x_1 = x_2, \\ 1 & \text{otherwise.} \end{cases}$$

(2) The Euclidean metric $d_R : \mathbb{R} \times \mathbb{R} \to [0, 1]$ is defined by

$$d_R(r_1, r_2) = \frac{|r_1 - r_2|}{1 + |r_1 - r_2|}.$$ 

(3) Let $A$ be a set. By $A^\infty$ we denote the set of finite and infinite sequences over $A$. The Baire metric $d_A^\infty : A^\infty \times A^\infty \to [0, 1]$ is defined by

$$d_A^\infty(\sigma_1, \sigma_2) = \begin{cases} 0 & \text{if } \sigma_1 = \sigma_2, \\ 2^{-n} & \text{otherwise,} \end{cases}$$

where $n$ is the length of the longest common prefix of $\sigma_1$ and $\sigma_2$.

A metric very similar to the one presented in Example A.2(3) was defined by Baire in [10]. Clearly, a nonempty set endowed with the discrete metric is a metric space.

Proposition A.3. $\mathbb{R}$ is a metric space.

Proof. Obviously, (1) and (2) are satisfied by $d_R$. We have left to prove that for all $r_1, r_2, r_3 \in \mathbb{R}$,

$$d_R(r_1, r_3) \leq d_R(r_1, r_2) + d_R(r_2, r_3).$$

We distinguish the following three cases.

- Let $|r_1 - r_3| \leq |r_1 - r_2|$. Then

$$d_R(r_1, r_3) = \frac{|r_1 - r_3|}{1 + |r_1 - r_3|} \leq \frac{|r_1 - r_2|}{1 + |r_1 - r_2|} = d_R(r_1, r_2) \leq d_R(r_1, r_2) + d_R(r_2, r_3).$$

- Let $|r_1 - r_3| \leq |r_2 - r_3|$. Similar to the previous case.
Exercise A.4. Prove that $A^\infty$ is a metric space.

Well-known operations on sets, like the Cartesian product $\times$, the disjoint union $+$, and the function space $\rightarrow$, can be lifted to metric spaces as follows.

Definition A.5. Let $X$ and $Y$ be metric spaces.

1. The metric $d_{X \times Y} : (X \times Y) \times (X \times Y) \rightarrow [0, 1]$ is defined by
   
   $d_{X \times Y}(\langle x_1, y_1 \rangle, \langle x_2, y_2 \rangle) = \max\{d_X(x_1, x_2), d_Y(y_1, y_2)\}$.

2. The metric $d_{X + Y} : (X + Y) \times (X + Y) \rightarrow [0, 1]$ is defined by
   
   $d_{X + Y}(v, w) = \begin{cases} 
   d_X(v, w) & \text{if } v, w \in X, \\
   d_Y(v, w) & \text{if } v, w \in Y, \\
   1 & \text{otherwise}.
   \end{cases}$

3. The metric $d_{X \rightarrow Y} : (X \rightarrow Y) \times (X \rightarrow Y) \rightarrow [0, 1]$ is defined by
   
   $d_{X \rightarrow Y}(f_1, f_2) = \sup_{x \in X} d_Y(f_1(x), f_2(x))$.

Note that the definition of the metric $d_{X \rightarrow Y}$ only relies on $d_Y$ (and not on $d_X$). The above introduced $d_{X \times Y}$, $d_{X + Y}$, and $d_{X \rightarrow Y}$ indeed give rise to a metric space. This is the contents of

Proposition A.6. If $X$ and $Y$ are metric spaces then so are $X \times Y$, $X + Y$, and $X \rightarrow Y$.

Proof. Easy. □

A.2. Completeness and contractiveness

The completeness of a metric space plays an important role in this paper. It is one of two ingredients of Banach’s theorem on which many of our results rely. The other
is the contractiveness of a function. Before we can define completeness, we first have to introduce the notions of convergent sequence and Cauchy sequence.

**Definition A.7.** Let \( X \) be a metric space.

1. A sequence \( (x_n) \) in \( X \) is **convergent** if
   \[
   \forall \varepsilon > 0: \exists N \in \mathbb{N}: \forall n \geq N: \quad d_X(x_n, x) \leq \varepsilon
   \]
   for some \( x \in X \).
2. A sequence \( (x_n) \) in \( X \) is **Cauchy** if
   \[
   \forall \varepsilon > 0: \exists N \in \mathbb{N}: \forall m, n \geq N: \quad d_X(x_m, x_n) \leq \varepsilon.
   \]

One can easily verify that a sequence converges to at most one element \( x \). This element (if it exists) is the **limit** of the sequence and is sometimes denoted by \( \lim_{n \to \infty} x_n \).

Clearly, if \( (x_n) \) converges to \( x \) then also every subsequence of \( (x_n) \) does so. Furthermore, observe that a subsequence of a Cauchy sequence is again Cauchy.

**Example A.8.**

1. Let \( X \) be a nonempty set endowed with the discrete metric. Every Cauchy sequence \( (x_n) \) in this space is eventually constant, that is,
   \[
   \exists N \in \mathbb{N}: \forall n \geq N: \quad x_n = x
   \]
   for some \( x \in X \), and converges to \( x \).
2. The sequence \( (2^{-n}) \) in \( \mathbb{R} \) is Cauchy and converges to 0.
3. The sequence \( (a^n b^n) \) in \( A^\infty \) is Cauchy and converges to \( d^0 \).
4. By \( A^* \) we denote the set of finite sequences over \( A \). The sequence \( (a^n b^n) \) in the metric space \( \langle A^*, d_{A^\infty} \rangle \) is Cauchy but not convergent.

We have the following

**Proposition A.9.** *Every convergent sequence is Cauchy.*

**Proof.** Let \( (x_n) \) be a convergent sequence in the metric space \( X \) with limit \( x \). We have to show that

\[
\forall \varepsilon > 0: \exists N \in \mathbb{N}: \forall m, n \geq N: \quad d_X(x_m, x_n) \leq \varepsilon.
\]

Let \( \varepsilon > 0 \). Since \( (x_n) \) converges to \( x \),

\[
\exists N \in \mathbb{N}: \forall n \geq N: \quad d_X(x_n, x) \leq \frac{\varepsilon}{2}.
\]

Let \( m, n \geq N \). Then

\[
d_X(x_m, x_n) = d_X(x_m, x) + d_X(x, x_n) \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2}.
\]
A metric space is complete if the converse of the above proposition also holds. Example A.8(4) is an example of a space for which this is not the case.

**Definition A.10.** A metric space is **complete** if each Cauchy sequence in the space is convergent.

**Example A.11.**
(1) Every nonempty set endowed with the discrete metric is complete.
(2) The metric space $A^*$ is not complete.

**Proposition A.12.** $\mathbb{R}$ is complete.

The proof of this proposition is postponed until Section A.3.

**Proposition A.13.** $A^\infty$ is complete.

**Proof.** Let $(\sigma_n)_n$ be a Cauchy sequence in $A^\infty$. We distinguish the following two cases.

Let the sequence $(\sigma_n)_n$ be eventually constant, that is,

$$\exists N \in \mathbb{N} \colon \forall n \geq N \colon \sigma_n = \sigma$$

for some $\sigma \in A^\infty$. Then $(\sigma_n)_n$ converges to $\sigma$.

Assume the sequence $(\sigma_n)_n$ is not eventually constant. Since $(\sigma_n)_n$ is Cauchy, without loss of any generality we can assume that

$$\forall n : d_{A^\infty}(\sigma_n, \sigma_{n+1}) \leq 2^{-n}.$$  

According to the definition of $d_{A^\infty}$, either $\sigma_n = \sigma_{n+1}$ or the length of the longest common prefix of $\sigma_n$ and $\sigma_{n+1}$ is at least $n$. Because $(\sigma_n)_n$ is not eventually constant, the length of $\sigma_n$ is at least $n$. Let $a_n$ be the $n$th element of $\sigma_n$. Then one can easily verify that $(\sigma_n)_n$ converges to $a_1a_2\cdots$. □

The operations on metric spaces introduced in Definition A.5 preserve completeness.

**Proposition A.14.** Let $X$ and $Y$ be metric spaces.

(1) If $X$ and $Y$ are complete then $X \times Y$ and $X + Y$ are complete.
(2) If $Y$ is complete then $X \to Y$ is complete.

**Proof.** We prove the first part of (1) as follows. Let $(x_n, y_n)_n$ be a Cauchy sequence in $X \times Y$. Then $(x_n)_n$ and $(y_n)_n$ are Cauchy sequences in $X$ and $Y$, respectively. Since $X$ and $Y$ are complete, $\lim_n x_n$ and $\lim_n y_n$ exist. The observation that $\lim_n (x_n, y_n) = (\lim_n x_n, \lim_n y_n)$ completes the proof. The second part can be proved similarly.

Let $(f_n)_n$ be a Cauchy sequence in $X \to Y$. Then $(f_n(x))_n$ is a Cauchy sequence for every $x \in X$. Since $Y$ is complete, $\lim_n f_n(x)$ exists. Next, we show that
\((f_n)_n\) converges to \(\lambda x. \lim_n f_n(x)\). Let \(\varepsilon > 0\). Since the sequence \((f_n)_n\) is Cauchy,
\[
\exists N \in \mathbb{N}: \forall m, n \geq N: \forall x \in X: d_Y(f_m(x), f_n(x)) \leq \frac{\varepsilon}{2}.
\]
Let \(n \geq N\) and \(x \in X\). Because \((f_n(x))_n\) converges to \(\lim_n f_n(x)\),
\[
\exists M \in \mathbb{N}: \forall m \geq M: d_Y(f_m(x), \lim_n f_n(x)) \leq \frac{\varepsilon}{2}.
\]
Consequently,
\[
d_Y(f_n(x), \lim_n f_n(x)) \\ \leq d_Y(f_n(x), f_{\max\{M,N\}}(x)) + d_Y(f_{\max\{M,N\}}(x), \lim_n f_n(x)) \\ \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2}.
\]

Next, we introduce contractive (and some other related) functions.

**Definition A.15.** Let \(X\) and \(Y\) be metric spaces. Let \(f : X \to Y\) be a function.

1. The function \(f\) is **continuous** if for every sequence \((x_n)_n\) converging to \(x\) we have that the sequence \((f(x_n))_n\) converges to \(f(x)\).
2. Let \(\alpha \geq 0\). The function \(f\) is **\(\alpha\)-Lipschitz** if for all \(x_1, x_2 \in X\),
   \[
d_Y(f(x_1), f(x_2)) \leq \alpha \cdot d_X(x_1, x_2).
   \]
3. A function is **nonexpansive** if it is 1-Lipschitz.
4. A function is **\(\alpha\)-contractive** if it is \(\alpha\)-Lipschitz for some \(0 \leq \alpha < 1\).

Note that every contractive function is nonexpansive. The converse does not hold (see Example A.16(2)).

**Example A.16.**

1. Let \(X\) and \(Y\) be nonempty sets endowed with the discrete metric. Every function \(f : X \to Y\) is nonexpansive.
2. The function \(f : [\mathbb{R}]\) defined by \(f(r) = r\) is nonexpansive but not contractive.
3. Let \(a \in A\). The function \(f : [A^\infty]\) defined by \(f(\sigma) = a\sigma\) is \(\frac{1}{2}\)-contractive.

**Exercise A.17.** Prove that every Lipschitz function is continuous.

After having introduced the notions of completeness and contractiveness, we now come to the main theorem of this appendix.

**Theorem A.18** (Banach [18]). Let \(X\) be a metric space and let \(f : [X]\) be a contractive function.

1. If \(f(x) = x\) and \(f(y) = y\) then \(x = y\).
2. If \(X\) is complete then for all \(x_0 \in X\), \(f(\lim_n x_n) = \lim_n x_n\), where \(x_{n+1} = f(x_n)\).
**Proof.** We prove (1) as follows. Assume \( f(x) = x \) and \( f(y) = y \). Since \( f \) is \( \alpha \)-contractive for some \( 0 \leq \alpha < 1 \),

\[
d_X(x, y) = d_X(f(x), f(y)) \leq \alpha \cdot d_X(x, y).
\]

Hence, \( d_X(x, y) = 0 \) which implies that \( x = y \). To verify (2) we first check that the sequence \( (x_n)_n \) is Cauchy. Let \( m < n \). Then

\[
d_X(x_m, x_n)
\leq \sum_{i=m}^{n-1} d_X(x_i, x_{i+1})
\leq \sum_{i=m}^{n-1} \alpha^i \cdot d_X(x_0, x_1) \quad [f \text{ is } \alpha\text{-contractive}]
= \left( \sum_{i=0}^{n-m-1} \alpha^i \right) \cdot \alpha^m \cdot d_X(x_0, x_1)
\leq \frac{\alpha^m}{1 - \alpha} \cdot d_X(x_0, x_1).
\]

Consequently, the sequence \( (x_n)_n \) is Cauchy. Since \( X \) is complete, its limit \( \lim_{n \to \infty} x_n \) exists. Furthermore, we have that

\[
f \left( \lim_{n \to \infty} x_n \right)
= \lim_{n \to \infty} f(x_n) \quad [\text{Exercise A.17}]
= \lim_{n \to \infty} x_{n+1}
= \lim_{n \to \infty} x_n. \quad \square
\]

The above theorem is known as Banach’s fixed point theorem. It tells us that a contractive function from a metric space to itself has at most one fixed point, and furthermore that a contractive function from a complete metric space to itself has a unique fixed point. For a contraction \( f \) from a complete metric space to itself we denote its unique fixed point by \( \text{fix}(f) \).

Based on the above theorem we have the following proof principle – the *unique fixed point proof principle* – to show that two elements of a metric space are equal. First, one introduces a function from the metric space to itself, and proves that the function is contractive. Then one shows that the elements to be shown equal are each a fixed point of the contraction. From Banach’s theorem we can now conclude that the two elements of the metric space are equal.

### A.3. Spaces of subsets

In this section, we study how the set \( \mathcal{P}_n(X) \) of nonempty subsets of a metric space \( X \) can be supplied with a suitable metric structure.
Definition A.19 (Hausdorff [30]). Let $X$ be a metric space. The Hausdorff metric

$$d_{\mathcal{P}(X)} : \mathcal{P}(X) \times \mathcal{P}(X) \to [0, 1]$$

is defined by

$$d_{\mathcal{P}(X)}(A, B) = \max \left\{ \sup_{a \in A} \inf_{b \in B} d_X(a, b), \sup_{b \in B} \inf_{a \in A} d_X(b, a) \right\}.$$

The above-introduced Hausdorff metric satisfies conditions (2) and (3) of Definition A.1 but it does not satisfy condition (1).

Exercise A.20. Let $a \in A$. Verify that

$$d_{\mathcal{P}(A^\infty)}(\{a^n | n \in \mathbb{N}\}, \{a^n | n \in \mathbb{N}\} \cup \{a\}) = 0.$$

To obtain a metric space, we do not consider all nonempty subsets of the metric space but only the nonempty and compact ones.

Definition A.21. Let $X$ be a metric space. A set $A \subseteq X$ is compact if every sequence in $A$ has a converging subsequence with its limit in $A$.

We usually denote a subsequence of a sequence $(x_n)_n$ by $(x_{s(n)})_n$, where $s : \mathbb{N} \to \mathbb{N}$ is a strictly increasing function.

Example A.22.

1. Every finite set is compact.
2. Let $X$ be a nonempty set endowed with the discrete metric. A subset of $X$ is compact if and only if it is finite.
3. The set $\{a^n | n \in \mathbb{N}\} \cup \{a\}$ is a compact subset of $A^\infty$ but $\{a^n | n \in \mathbb{N}\}$ is not.

Proposition A.23. Closed intervals of $\mathbb{R}$ are compact.

Proof. See, for example, [51].

Now we can present

Proof of Proposition A.12. Let $(r_n)_n$ be a Cauchy sequence in $\mathbb{R}$. Then

$$\exists N \in \mathbb{N}: \forall n \geq N: d_{\mathbb{R}}(r_n, r_N) \leq \frac{1}{2}.$$ 

Hence, for all $n \geq N$, we have that $r_n \in [r_N - 1, r_N + 1]$. Since the closed interval $[r_N - 1, r_N + 1]$ is compact (Proposition A.23), the sequence $(r_{N+n})_n$ has a subsequence converging to some $r \in [r_N - 1, r_N + 1]$. Consequently, the Cauchy sequence $(r_n)_n$ converges to $r$. 

For nonempty and compact subsets of a metric space we have the following
Proposition A.24. Let \( X \) be a metric space. Let \( A \) and \( B \) be nonempty and compact subsets of \( X \). Then

\[
\sup_{a \in A} \inf_{b \in B} d_X(a, b) = \max_{a \in A} \min_{b \in B} d_X(a, b).
\]

Proof. Let \( a \in A \). We only show that \( \inf_{b \in B} d_X(a, b) = \min_{b \in B} d_X(a, b) \). The rest of the proof can be completed similarly. Clearly,

\[
\forall n \in \mathbb{N} \colon \exists b_n \in B: d_X(a, b_n) \leq \inf_{b \in B} d_X(a, b) + 2^{-n}. \tag{A.1}
\]

Since the set \( B \) is compact, the sequence \( (b_n)_n \) has a subsequence \( (b_{s(n)})_n \) converging to \( \lim_n b_{s(n)} \in B \). We claim that \( d_X(a, \lim_n b_{s(n)}) = \inf_{b \in B} d_X(a, b) \). To prove this it suffices to show that

\[
\forall n \in \mathbb{N} \colon d_X\left(a, \lim_n b_{s(n)}\right) \leq \inf_{b \in B} d_X(a, b) + 2^{-n+1}.
\]

Let \( n \in \mathbb{N} \). Then

\[
\exists M \in \mathbb{N} \colon \forall m \geq M: d_X\left(b_{s(m)}, \lim_n b_{s(n)}\right) \leq 2^{-n}. \tag{A.2}
\]

Hence,

\[
d_X\left(a, \lim_n b_{s(n)}\right) \\
\leq d_X(a, b_{s(M)}) + d_X\left(b_{s(M)}, \lim_n b_{s(n)}\right) \\
\leq \inf_{b \in B} d_X(a, b) + 2^{-n} + 2^{-n} \quad \text{[(A.1) and (A.2)].}
\]

As a consequence of the above proposition, the Hausdorff metric restricted to the set \( \mathcal{P}_{nc}(X) \) of nonempty and compact subsets of a metric space \( X \)

\[
d_{\mathcal{P}_{nc}(X)}: \mathcal{P}_{nc}(X) \times \mathcal{P}_{nc}(X) \to [0, 1]
\]

amounts to

\[
d_{\mathcal{P}_{nc}(X)}(A, B) = \max \left\{ \max_{a \in A} \min_{b \in B} d_X(a, b), \max_{b \in B} \min_{a \in A} d_X(b, a) \right\}.
\]

From the above formulation immediately follows that for all \( \varepsilon \geq 0 \), we have that \( d_{\mathcal{P}_{nc}(X)}(A, B) \leq \varepsilon \) if and only if

\[
\forall a \in A: \exists b \in B: d_X(a, b) \leq \varepsilon \quad \text{and} \quad \forall b \in B: \exists a \in A: d_X(b, a) \leq \varepsilon.
\]

This fact is exploited in the proof of

Proposition A.25. Let \( X \) be a metric space. Then \( \mathcal{P}_{nc}(X) \) is a metric space.
Proof. First of all,
\[ d_{\mathcal{Pnc}(X)}(A, B) = 0 \]
iff \( \forall a \in A: \exists b \in B: d_X(a, b) = 0 \)
and \( \forall b \in B: \exists a \in A: d_X(b, a) = 0 \)
iff \( A \subseteq B \) and \( B \subseteq A \)
iff \( A = B \).

Obviously, \( d_{\mathcal{Pnc}(X)}(A, B) = d_{\mathcal{Pnc}(X)}(B, A) \).

We have left to prove that \( d_{\mathcal{Pnc}(X)}(A, C) \leq d_{\mathcal{Pnc}(X)}(A, B) + d_{\mathcal{Pnc}(X)}(B, C) \). It suffices to show that
\[ \forall a \in A: \exists c \in C: d_X(a, c) \leq d_{\mathcal{Pnc}(X)}(A, B) + d_{\mathcal{Pnc}(X)}(B, C) \]
and
\[ \forall c \in C: \exists a \in A: d_X(c, a) \leq d_{\mathcal{Pnc}(X)}(A, B) + d_{\mathcal{Pnc}(X)}(B, C) \].

We only prove the first part. The second part can be proved similarly. Let \( a \in A \).
Then we have that \( d_X(a, b) \leq d_{\mathcal{Pnc}(X)}(A, B) \) for some \( b \in B \). Also we have that \( d_X(b, c) \leq d_{\mathcal{Pnc}(X)}(B, C) \) for some \( c \in C \). Hence,
\[ d_X(a, c) \leq d_X(a, b) + d_X(b, c) \leq d_{\mathcal{Pnc}(X)}(A, B) + d_{\mathcal{Pnc}(X)}(B, C) \].

Now, we have another operation on metric spaces: taking its nonempty and compact subsets and endowing them with the Hausdorff metric. This operation preserves completeness.

Theorem A.26 (Kuratowski [37]). If \( X \) is complete then \( \mathcal{Pnc}(X) \) is complete.

Proof. Let \((A_n)\) be a Cauchy sequence in \( \mathcal{Pnc}(X) \). Without any loss of generality we can assume that for all \( n \),
\[ d_{\mathcal{Pnc}(X)}(A_n, A_{n+1}) \leq 2^{-n}. \] (A.3)

We will show that the set
\[ A = \left\{ \lim_n a_n \mid a_n \in A_n \text{ and } (a_n)_n \text{ is a Cauchy sequence} \right\} \]
is nonempty and compact, and that it is the limit of the sequence.

First, we show that the set \( A \) is nonempty by inductively constructing a sequence \((a_n)_n\), with \( a_n \in A_n \), which is Cauchy. Let \( a_0 \) be an arbitrary element of \( A_0 \). Having chosen \( a_0, \ldots, a_n \) we pick \( a_{n+1} \in A_{n+1} \) such that \( d_X(a_n, a_{n+1}) \leq 2^{-n} \) which is possible according to Eq. (A.3). We have left to prove that the constructed sequence is Cauchy.
This follows from the fact that for \( m \leq n \) we have that
\[
\begin{align*}
   d_X(a_m, a_n) \\
   \leq \sum_{i=m}^{n-1} d_X(a_i, a_{i+1}) \\
   \leq \sum_{i=m}^{n-1} 2^{-i} \\
   \leq 2^{-m+1}.
\end{align*}
\]

Second, we prove that the set \( A \) is compact. Let \((\bar{a}_m)_m\) be a sequence in \( A \). For all \( m \), we have that \( \bar{a}_m = \lim_n a_{m,n} \) for some Cauchy sequence \((a_{m,n})_n\) with \( a_{m,n} \in A_n \). Without any loss of generality we can assume that for all \( m \) and \( n \),
\[
d_X(a_{m,n}, \bar{a}_m) \leq 2^{-n+2}
\]
(see Exercise A.27).

We have to show that \((\bar{\bar{a}}_m)_m\) has a converging subsequence. This is proved as follows. First, we construct for each sequence \((a_{m,n})_m\) a suitable subsequence \((a_{s_{m,n}})_m\). These subsequences give us the subsequence of \((\bar{a}_m)_m\) we are after: \((\bar{\bar{a}}_{s_{m,n}})_m\). Second, we show that the sequence \((a_{s_{m,n}})_n\), with \( a_{s_{m,n}} \in A_n \), is Cauchy, and hence its limit \( \bar{\bar{a}} \) is an element of \( A \). Finally, we prove that \( \bar{\bar{a}} \) is the limit of the subsequence \((\bar{\bar{a}}_{s_{m,n}})_m\).

We start with inductively constructing the subsequences \((a_{s_{m,n}})_m\). Since the set \( A_0 \) is compact, the sequence \((a_{m,0})_m\) has a converging subsequence \((a_{s_{m,0}})_m\) such that for all \( m \),
\[
\forall i \geq m: \ d_X(a_{s_{m,0}}, a_{s_{i,0}}) \leq 2^{-m}.
\]
Having constructed the subsequences \((a_{s_0(m)})_m, \ldots, (a_{s_n(m),n})_m\), we build the subsequence \((a_{s_{n+1}(m),n+1})_m\) as follows. Because the set \(A_{n+1}\) is compact, the sequence \((a_{s_n(m),n+1})_m\) has a converging subsequence \((a_{s_{n+1}(m),n+1})_m\) satisfying for all \(m\),

\[
\forall i \geq m : d_X(a_{s_{n+1}(m),n+1}, a_{s_{n+1}(i),n+1}) \leq 2^{-m}. \tag{A.5}
\]

Next, we consider the sequence \((a_{s_n(n)})_n\). Since for all \(n\),

\[
d_X(a_{s_{n+1}(n),n}, a_{s_{n+1}(n+1),n}) \leq d_X(a_{s_{n+1}(n+1),n}, \tilde{a}_{s_{n+1}(n+1)}) + d_X(\tilde{a}_{s_{n+1}(n+1)}, a_{s_{n+1}(n+1),n}) \\
\leq 2^{-n} + 2^{-n+2} + 2^{-n+1} \quad \text{[Eqs. (A.4) and (A.5)]} \\
\leq 2^{-n+3}, \tag{A.6}
\]

we can conclude that the sequence is Cauchy. Because for all \(m\) and \(n\),

\[
d_X(\tilde{a}_{s_n(m)}, a_{s_{n+1}(m+n),m+n}) \leq d_X(\tilde{a}_{s_n(m)}, a_{s_{n+1}(m),m}) + \sum_{i=m}^{m+n-1} d_X(a_{s(i),i}, a_{s(i+1),i+1}) \\
\leq 2^{-m+2} + \sum_{i=m}^{m+n-1} 2^{-i+3} \quad \text{[Eqs. (A.4) and (A.6)]} \\
\leq 2^{-m+5},
\]

we can derive from the fact stated in Exercise 33 that for all \(m\),

\[
d_X(\tilde{a}_{s_n(m)}, \lim_n a_{s_n(n)}) = d_X(\tilde{a}_{s_n(m)}, \lim_n a_{s_{n+1}(m+n),m+n}) \leq 2^{-m+5}. \tag{A.7}
\]

Hence, the subsequence \((\tilde{a}_{s_n(m)})_m\) converges to \(\tilde{a}\). We conclude this proof by showing that \(A\) is the limit of the sequence \((A_n)_n\). It suffices to prove that for all \(n\),

\[
\forall a \in A : \exists a_n \in A_n : d_X(\tilde{a}, a_n) \leq 2^{-n+2} \tag{A.7}
\]

and

\[
\forall a_n \in A_n : \exists \tilde{a} \in A : d_X(a_n, \tilde{a}) \leq 2^{-n+1}. \tag{A.8}
\]

We start with demonstrating the validity of Eq. (A.7). Let \(a \in A\). By definition, there exists a Cauchy sequence \((a'_{m})_m\), with \(a'_{m} \in A_{m}\), which converges to \(a\). Consequently,

\[
\exists M \in \mathbb{N} : \forall m \geq M : d_X(a'_{m}, \tilde{a}) \leq 2^{-n+1}.
\]

According to Eq. (A.3), there exists an \(a_n \in A_n\) such that

\[
d_X(a_n, a'_{\max(M,n)}) \leq 2^{-n+1}.
\]

Combining the above we arrive at Eq. (A.7). We still have to verify Eq. (A.8). Let \(a_n \in A_n\). We inductively construct a Cauchy sequence \((a'_{m})_m\), with \(a'_{m} \in A_{m}\), and
show that its limit $\tilde{a}$ satisfies Eq. (A.8). We choose $a'_1 \in A_1, \ldots, a'_{n-1} \in A_{n-1}$ arbitrarily. We take $a'_n$ to be $a_n$. Having chosen $a'_1, \ldots, a'_{n+m}$, we pick $a'_{n+m+1} \in A_{n+m+1}$ such that $d_X(a'_{n+m}, a'_{n+m+1}) \leq 2^{-(n+m)}$ which is possible according to Eq. (A.3). Since for all $m$,

$$d_X(a_n, a'_n) \leq d_X(a_n, a'_n) + d_X(a'_n, a'_{n+m}) \leq 2^{-n+1},$$

we can deduce from the fact given in Exercise 33 that

$$d_X\left(a_n, \lim_m a'_m\right) = d_X\left(a_n, \lim_m a'_{n+m}\right) \leq 2^{-n+1}.$$

**Exercise A.27.** Complete the proof of Theorem A.26.

To conclude that a set is compact, the following theorem can be useful.

**Theorem A.28** (Michael [41]). Let $X$ be a metric space.

1. If $\mathcal{A} \subseteq \mathcal{P}_{nc}(X)$ then $\bigcup \mathcal{A} \in \mathcal{P}_{nc}(X)$.
2. The function $\bigcup : \mathcal{P}_{nc}(X) \to \mathcal{P}_{nc}(X)$ is nonexpansive.

**Proof.** Let $(x_n)_n$ be a sequence in $\bigcup \mathcal{A}$. Then there exists a sequence $(A_n)_n$ in $\mathcal{A}$ satisfying $x_n \in A_n$. Because $\mathcal{A}$ is compact, $(A_n)_n$ has a subsequence $(A'_n)_n$ converging to some $A \in \mathcal{A}$. For each $x_n$ there exists a $y_n \in A$ such that $d_X(x_n, y_n) \leq d_{\mathcal{P}_{nc}(X)}(A_n, A)$. Since $A$ is compact, the sequence $(y_{n_{(n)})})_{n}$ has a subsequence $(y'_{n_{(n)})})_{n}$ converging to some $y \in A$. Because

$$d_X(x_{n_{(n)}}, y) \leq d_X(x_{n_{(n)}}, y_{n_{(n)})}) + d_X(y_{n_{(n)}}, y) \leq d_{\mathcal{P}_{nc}(X)}(A'_{n_{(n)}}, A) + d_X(y_{n_{(n)}}, y),$$

the sequence $(x_n)_n$ has a subsequence $(x'_{n_{(n)})})_n$ converging to $y \in \bigcup \mathcal{A}$.

The proof of the second part is left to the reader as Exercise A.29.

**Exercise A.29.** Prove the second part of Theorem A.28.

We add the empty set by defining

$$\mathcal{P}_e(X) = \mathcal{P}_{nc}(X) + \{\emptyset\},$$

where $\{\emptyset\}$ is a singleton metric space.

**A.4. Nonexpansive functions**

We already introduced nonexpansive functions in Definition A.15(3). They play an important role in this paper, mainly because they preserve compactness (see Theorem A.31).

By $X \longrightarrow Y$ we denote the set of all nonexpansive functions from the metric space $X$ to the metric space $Y$. We endow this set with the metric introduced in
Definition A.5(3) restricted to the nonexpansive functions. Obviously, this gives us a metric space.

**Proposition A.30.** If $Y$ is complete then $X \rightarrow_1 Y$ is complete.

**Proof.** Let $(f_n)_n$ be a Cauchy sequence of nonexpansive functions. Let $x_1, x_2 \in X$. For all $n$, we have that $d_Y(f_n(x_1), f_n(x_2)) \leq d_X(x_1, x_2)$. According to the fact stated in Exercise 33,

$$d_Y \left( \lim_n f_n(x_1), \lim_n f_n(x_2) \right) \leq d_X(x_1, x_2).$$

The observation made in the proof of Proposition A.14 that $(f_n)_n$ converges to $\lambda x. \lim_n f_n(x)$ completes the proof. 

As we already mentioned above, the nonexpansive image of a compact set is compact.

**Theorem A.31** (Alexandroff [1]). Let $X$ and $Y$ be metric spaces. Let $f : X \rightarrow Y$ be a nonexpansive function. If $A$ is a compact subset of $X$ then $\{ f(x) \mid x \in A \}$ is a compact subset of $Y$.

**Proof.** Let $(f(x_n))_n$ be a sequence in the set. Then $(x_n)_n$ is a sequence in $A$. Since $A$ is compact, the sequence $(x_n)_n$ has a subsequence $(x_{s(n)})_n$ converging to some $x \in A$. Since $f$ is nonexpansive, and hence continuous (Exercise A.17), the subsequence $(f(x_{s(n)}))_n$ converges to $f(x)$. 

**Appendix B. Labelled transition systems**

In this appendix, we study operational semantic models defined by means of labelled transition systems. A labelled transition system has a collection of configurations. These configurations are usually statements possibly decorated with some additional information, for example, the values of the variables.

$$\begin{bmatrix} x := v; u := w; w := z, \begin{bmatrix} v = 1 \\ w = 4 \\ x = 2 \end{bmatrix} \end{bmatrix}$$

The computation steps of a statement are described by means of transitions. A transition brings the system from one configuration into another one. The transitions are labelled by actions. The label of a transition tells us something about the computation step, for example, the assignment of a value to a variable.

$$\begin{bmatrix} x := v; u := w; w := z, \begin{bmatrix} v = 1 \\ w = 4 \\ x = 2 \end{bmatrix} \end{bmatrix} \xrightarrow{z := 1} \begin{bmatrix} u := w; u := z, \begin{bmatrix} v = 1 \\ w = 4 \\ x = 1 \end{bmatrix} \end{bmatrix}$$
The use of labelled transition systems to give semantics seems to originate with Keller [33].

We consider both terminating and nonterminating labelled transition systems. A system is nonterminating if there exists a (countably) infinite sequence of subsequent transitions.

\[
\text{while true do } v := v + 1 \text{ od, } \langle v = 1 \rangle \\
v := 2 \\
\text{while true do } v := v + 1 \text{ od, } \langle v = 2 \rangle \\
v := 3 \\
\vdots
\]

We distinguish between deterministic and nondeterministic labelled transition systems. Nondeterminism arises when configurations have multiple outgoing transitions. Reaching the configuration \( [v := 1 \| v := 2, \langle v = 0 \rangle] \), the computation evolves nondeterministically by either doing the left assignment first and the right one second or vice versa.

We focus on a restricted form of nondeterminism. This degree of nondeterminism is described in terms of the branching degree of labelled transition systems. A labelled transition system is \textit{finitely branching} if every configuration has only finitely many outgoing transitions. Most nondeterministic languages can be described by means of a finitely branching labelled transition system.

From a labelled transition system one can derive an operational semantics in various ways. We will extract from the system a function assigning to each configuration of the system a (finite or infinite) sequence of actions or a set of (finite and infinite) sequences of actions. Given a labelled transition system, one can also assign other structures to the configurations of the system. See, for example, [48] for some other structures than (sets of) sequences.

We develop some theory to prove these operational semantic models equal to other semantic models. These proofs are based on the \textit{unique fixed point proof principle}. By means of this proof principle, elements of a metric space can be shown to be equal. First, one introduces a function from the metric space to itself, and proves that the function is a contraction. Then one shows that the elements to be shown equal are each a fixed point of the contraction. To apply this proof principle to show that semantic models are equal, the models should be elements of a metric space. Furthermore, a contractive function from the metric space to itself with the semantic models as fixed
point is needed. To fulfill the second requirement – the existence of a contractive function from the metric space to itself with the operational semantics as fixed point – we introduce **semantics transformations**. A semantics transformation is a function from a space of semantic models to itself. Like the operational semantic models, also the semantics transformations are defined by means of labelled transition systems. The space of semantic models is the collection of functions from the set of configurations of the system to the set of actions sequences or the set of (restricted) sets of actions sequences, supplied with a suitable metric. Semantic transformations were first studied in an order-theoretic setting by Hennessy and Plotkin [32]. Kuiper [36] and De Bruin [21] used them in a metric setting. Metric semantic transformations were exploited in a systematic way by Kok and Rutten in [34].

In this appendix we bring together a number of results scattered over the literature. In Section B.1, we introduce labelled transition systems. In Section B.2, we show how to derive an operational semantics from these systems. Furthermore, these models are proved to be an element of a metric space – the first ingredient to apply the unique fixed point proof principle. The second ingredient, the semantics transformations, are studied in Section B.3. The examples we present in this section have been chosen as simple as possible. The configurations and the actions of the labelled transition systems are uninterpreted. In the main text, we illustrate how labelled transition systems can be used to define the operational semantics of programming and specification languages.

### B.1. Labelled transition systems

In this section, we introduce labelled transition systems and some related notions.

**Definition B.1.** A labeled transition system is a triple \( \langle C, A, \rightarrow \rangle \) consisting of
- a nonempty set of configurations \( C \),
- a nonempty set of actions \( A \), and
- a (labelled) transition relation \( \rightarrow \subseteq C \times A \times C \).

Instead of \( \langle c, a, c' \rangle \in \rightarrow \) we write \( c \xrightarrow{a} c' \). Most of the time we only present the transition relation of the labelled transition system.

**Example B.2.** The labelled transition system

\[
\langle \{c_1, c_2\}, \{a_1, a_2\}, \{\langle c_1, a_1, c_1 \rangle, \langle c_1, a_2, c_2 \rangle \} \rangle
\]

is presented by

\[
\begin{align*}
  c_1 & \xrightarrow{a_1} c_1, \\
  c_1 & \xrightarrow{a_2} c_2.
\end{align*}
\]

In the examples we assume the configurations, like \( c_1 \) and \( c_2 \), and the actions, like \( a_1 \) and \( a_2 \), all to be different.
If \( c \xrightarrow{a} c' \) then we say that there exists a transition from \( c \) to \( c' \) labelled by \( a \). If there exists a transition from \( c \), we write \( c \rightarrow \). Otherwise, we write \( c \rightarrow^* \). By means of these predicates \( \rightarrow \) and \( \rightarrow^* \) we partition the set of configurations into the sets of nonterminal and terminal configurations.

**Definition B.3.** A configuration \( c \) is **nonterminal** if \( c \rightarrow \) and it is **terminal** if \( c \rightarrow^* \).

**Example B.4.** In Example B.2, \( c_1 \) is a nonterminal configuration and \( c_2 \) is a terminal configuration.

Frequently, we depict (the transition relation of) a labelled transition system by a directed graph. The nodes are labelled by configurations and the edges are indexed by actions.

**Example B.5.** The labelled transition system introduced in Example B.2 can be depicted by

\[
\begin{array}{ccc}
& a_1 & \\
\downarrow & & \\
C_1 & \xrightarrow{a_2} & C_2
\end{array}
\]

A labelled transition system is nonterminating if the corresponding graph contains a (countably) infinite path.

**Definition B.6.** A labelled transition system \( \langle C, A, \rightarrow \rangle \) is **nonterminating** if there exist \( c_0, c_1, \ldots \in C \) and \( a_1, a_2, \ldots \in A \) such that, for all \( n \in \mathbb{N} \),

\[
c_n \xrightarrow{a_{n+1}} c_{n+1}.
\]

**Example B.7.** The labelled transition system introduced in Example B.2 is nonterminating. If we leave out the transition \( c_1 \xrightarrow{a_1} c_1 \), then we obtain the terminating system depicted by

\[
c_1 \xrightarrow{a_2} c_2
\]

A labelled transition system is deterministic if every configuration has at most one outgoing transition.

**Definition B.8.** A labelled transition system \( \langle C, A, \rightarrow \rangle \) is **deterministic** if, for all \( c \in C \), the set

\[\mathcal{F}(c) = \{ (a, c') \mid c \xrightarrow{a} c' \}\]

contains at most one element.

**Example B.9.** The labelled transition system introduced in Example B.2 is nondeterministic. If we erase the transition \( c_1 \xrightarrow{a_1} c_2 \), then we get the deterministic system
depicted by

We conclude this section with the definition of a restricted degree of nondeterminism. A labelled transition system is finitely branching if every configuration has only finitely many outgoing transitions.

**Definition B.10.** A labelled transition system \( \langle C, A, \rightarrow \rangle \) is *finitely branching* if, for all \( c \in C \), the set \( \mathcal{F}(c) \) is finite.

**Example B.11.** The labelled transition system defined in Example B.2 is finitely branching. However, the labelled transition system

\[
    c \xrightarrow{a_n} c_n \quad \text{for } n \in \mathbb{N}
\]

depicted by

is not.

**B.2. Operational semantics**

Given a labelled transition system we can derive from it an operational semantics in various ways. We extract from the system a function assigning to each configuration an action sequence or a set of action sequences. We focus on deterministic and terminating, deterministic and nonterminating, and nondeterministic and nonterminating systems.

**B.2.1. Deterministic and terminating**

We start with deterministic and terminating labelled transition systems. For such a system, the operational semantics assigns to each configuration a finite sequence of actions. This sequence corresponds to the labels of the maximal transition sequence starting from the configuration.

**Definition B.12.** Let \( \langle C, A, \rightarrow \rangle \) be a deterministic and terminating labelled transition system. The *operational semantics* induced by \( \langle C, A, \rightarrow \rangle \) is the function \( \mathcal{O} : C \rightarrow A^* \) defined by

\[
    \mathcal{O}(c) = a_1a_2\cdots a_n \quad \text{if } c = c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} c_n \xrightarrow{}.
\]

In the above definition we use

\[
    c = c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} c_n \xrightarrow{}
\]
as an abbreviation for
\[ c = c_0 \land \forall 0 \leq m < n: c_m \xrightarrow{a_{m+1}} c_{m+1} \land c_n \xrightarrow{e}. \]

The operational semantics of the configuration \( c \) is the sequence \( a_1a_2 \cdots a_n \) if there exists a transition sequence from \( c \) to some terminal configuration labelled by \( a_1a_2 \cdots a_n \).

Note that the operational semantics of a terminal configuration is the empty sequence \( e \).

**Example B.13.** The operational semantics induced by the labelled transition system introduced in Example B.7 is given by

\[
\begin{align*}
\mathcal{O}(c_1) &= a_2 \\
\mathcal{O}(c_2) &= e.
\end{align*}
\]

**B.2.2. Deterministic and nonterminating**

The operational semantics induced by a deterministic and nonterminating labelled transition system assigns to each configuration a finite or an infinite sequence of actions.

**Definition B.14.** Let \( \langle C, A, \rightarrow \rangle \) be a deterministic and nonterminating labelled transition system. The *operational semantics* induced by \( \langle C, A, \rightarrow \rangle \) is the function \( \mathcal{O} : C \rightarrow A^\infty \) defined by

\[
\mathcal{O}(c) = \begin{cases} 
    a_1a_2 \cdots a_n & \text{if } c = c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} c_n \xrightarrow{e} \\
    a_1a_2 \cdots & \text{if } c = c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} \cdots
\end{cases}
\]

In the above definition we use

\[ c = c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} \cdots \]

as an abbreviation for

\[ c = c_0 \land \forall m \geq 0: c_m \xrightarrow{a_{m+1}} c_{m+1}. \]

If there is a (countably) infinite transition sequence from \( c \) labelled by \( a_1a_2 \cdots \), then the infinite action sequence \( a_1a_2 \cdots \) is the operational semantics of the configuration \( c \).

**Example B.15.** The operational semantics induced by the labelled transition system introduced in Example B.9 is given by

\[ \mathcal{O}(c_1) = a_1^0. \]

The set \( A^\infty \) of finite and infinite action sequences endowed with the Baire metric (Example A.2(3)) gives us a metric space. According to Definition A.5(3), we can also turn \( C \rightarrow A^\infty \) into a metric space. Hence, the operational semantic models induced by deterministic and nonterminating labelled transition systems are element of a metric space – one of the ingredients to apply the unique fixed point proof principle.
B.2.3. Nondeterministic and nonterminating

All labelled transition systems considered in the rest of this section are nondeterministic and nonterminating. For those systems, the operational semantics assigns to each configuration a nonempty set of finite or infinite action sequences.

**Definition B.16.** The operational semantics induced by the labelled transition system \( \langle C, A, \rightarrow \rangle \) is the function \( \mathcal{O} : C \rightarrow \mathcal{P}(A^\infty) \) defined by

\[
\mathcal{O}(c) = \{a_1a_2\cdots a_n \mid c = c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} c_n \} \cup \\
\{a_1a_2\cdots \mid c = c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} \cdots \}.
\]

A sequence \( a_1a_2\cdots a_n \) is an element of the operational semantics of the configuration \( c \) if there exists a transition sequence from \( c \) to some terminal configuration labelled by \( a_1a_2\cdots a_n \). If there exists an infinite transition sequence from \( c \) labelled by \( a_1a_2\cdots \), then the infinite sequence \( a_1a_2\cdots \) is an element of the operational semantics of \( c \). The operational semantics of a terminal configuration is a singleton set consisting of the empty sequence \( \varepsilon \).

**Example B.17.** The labelled transition system introduced in Example B.2 induces the operational semantics \( \mathcal{O} \) defined by

\[
\mathcal{O}(c_1) = \{a_1^na_2 \mid n \in \mathbb{N}\} \cup \{a_1^\omega\}, \\
\mathcal{O}(c_2) = \{\varepsilon\}.
\]

To prove an operational semantics to be equal to another semantic model by means of the unique fixed point proof principle, we have to ensure that the operational semantics is an element of a metric space. We endow the set \( A^\infty \) of finite and infinite action sequences with the Baire metric. The set \( \mathcal{P}_n(A^\infty) \) of nonempty sets of these sequences is provided with the induced Hausdorff metric (see Definition A.19). As we have already seen in Section A.3, we do not obtain a metric space. By restricting ourselves to the subspace \( \mathcal{P}_{nc}(A^\infty) \) of nonempty and compact sets of action sequences we do get a metric space. Definition A.5(3) tells us how to provide \( C \rightarrow \mathcal{P}_{nc}(A^\infty) \) with a metric structure. We will focus on operational semantic models that belong to this metric space.

**Definition B.18.** An operational semantics \( \mathcal{O} : C \rightarrow \mathcal{P}(A^\infty) \) is compact if \( \mathcal{O} \in C \rightarrow \mathcal{P}_{nc}(A^\infty) \).

**Example B.19.** The operational semantics presented in Example B.17 is compact.

**Exercise B.20.** Demonstrate that not every labelled transition system induces a compact operational semantics.
As we have already seen above, not every labelled transition system induces a compact operational semantics. However, if we restrict ourselves to finitely branching labelled transitions systems, then we do obtain compact operational semantic models.

**Theorem B.21.** The operational semantics induced by a finitely branching labelled transition system is compact.

**Proof.** Let \( \langle C, A, \rightarrow \rangle \) be a finitely branching labelled transition system. We prove that the induced operational semantics \( O \) is compact, that is, for all \( c \in C \), the set \( O(c) \) is compact.

Let \( c \in C \). Let \( (\sigma_n)_n \) be a sequence in \( O(c) \). We show that there exists a subsequence \( (\sigma_{n_k})_k \) of \( (\sigma_n)_n \) converging to some \( \sigma \in O(c) \).

The subsequence \( (\sigma_{n_k})_k \) will be constructed from a collection of subsequences \( (\sigma_{n_k})_k \) satisfying

\[
(\forall m \in \mathbb{N}: Q(m)) \lor (\exists k \in \mathbb{N}: \forall 1 \leq m < k: Q(m) \land R(k)) \quad \text{(B.10)}
\]

where

\[
Q(m) \iff \forall n \in \mathbb{N}: \sigma_{n_k} = a_1 a_2 \cdots a_m \sigma_m, (n_k) \land c = c_0 \overset{a_1}{\rightarrow} c_1 \overset{a_2}{\rightarrow} \cdots \overset{a_m}{\rightarrow} c_m \rightarrow \land \\
\sigma_m, (n_k) \in O(c_m)
\]

and

\[
R(m) \iff \forall n \in \mathbb{N}: \sigma_{n_k} = a_1 a_2 \cdots a_m \land c = c_0 \overset{a_1}{\rightarrow} c_1 \overset{a_2}{\rightarrow} \cdots \overset{a_m}{\rightarrow} c_m \rightarrow .
\]

The existence of the subsequences \( (\sigma_{n_k})_k \) is verified by proving that, for all \( i \in \mathbb{N} \),

\[
P(i) \iff (\forall 1 \leq m \leq i: Q(m)) \lor (\exists 1 \leq k \leq i: \forall 1 \leq m < k: Q(m) \land R(k))
\]

by induction on \( i \).

To prove \( P(0) \) it suffices to show \( Q(0) \lor R(0) \). Clearly, the sequence \( (\sigma_n)_n \) satisfies \( Q(0) \lor R(0) \).

Let \( i > 0 \). To prove \( P(i - 1) \Rightarrow P(i) \) it suffices to show \( Q(i - 1) \Rightarrow Q(i) \lor R(i) \). If \( Q(i - 1) \), then

\[
(\sigma_{n_{i-1}(n)} = a_1 a_2 \cdots a_{i-1} a_{i-1} a_{i-1} \sigma_{i-1}(n) \sigma_{i-1}(n)) \land \\
c = c_0 \overset{a_1}{\rightarrow} c_1 \overset{a_2}{\rightarrow} \cdots \overset{a_{i-1}}{\rightarrow} c_{i-1} \overset{a_i}{\rightarrow} c_{i-1} \rightarrow \land \\
\sigma_{i-1}(n) \in O(c_{i-1}(n)) \lor \\
(\sigma_{n_{i-1}(n)} = a_1 a_2 \cdots a_{i-1} a_{i-1} \sigma_{i-1}(n) \land \\
c = c_0 \overset{a_1}{\rightarrow} c_1 \overset{a_2}{\rightarrow} \cdots \overset{a_{i-1}}{\rightarrow} c_{i-1} \overset{a_i}{\rightarrow} c_{i-1} \rightarrow )).
\]
Because the labelled transition system is finitely branching, there are only finitely many \((a_{i, s_{i-1}(n)}, c_{i, s_{i-1}(n)})\)'s satisfying

\[ c_{i-1} \xrightarrow{a_{i, s_{i-1}(n)}} c_{i, s_{i-1}(n)}. \]

Consequently, there exists a subsequence \((s_{m(n)})_n\) such that \(a_{i, s_{m(n)}} = a_i\) and \(c_{i, s_{m(n)}} = c_i\) for some \(a_i \in A\) and \(c_i \in C\). If \(c_i \rightarrow\), then \(Q(i)\) is satisfied. Otherwise, \(R(i)\) holds.

From the subsequences \((s_{m(n)})_n\) satisfying Eq. (B.10) we construct the subsequence \((s_{n})_n\) distinguishing the following two cases.

1. If \(\forall m \in \mathbb{N}: Q(m)\), then we define \(s_{n} = s_{n(n)}\). In this case, the sequence \((s_{n(n)})_n\) converges to \(s = a_1 a_2 \cdots\) in \(\ell(c)\).

2. If \(\exists k \in \mathbb{N}: 1 \leq m < k: Q(m) \land R(k)\), then we define \(s = s_{k}\). The sequence \((s_{n(n)})_n\) converges to \(s = a_1 a_2 \cdots a_k\) in \(\ell(c)\).

The above theorem is reminiscent to König’s lemma [35]. Related results have been presented by Arnold [5], De Bakker and Kok [12], De Bakker et al. [13] and Landweber [38].

As a consequence of the above theorem, the operational semantics induced by a finitely branching labelled transition system is an element of a metric space. We will exploit this fact to prove the operational model to be equal to another model by uniqueness of fixed point.

**B.3. Semantics transformations**

As we have already seen in Section B.2, the operational semantics induced by a deterministic and nonterminating labelled transition system or a finitely branching (nondeterministic) and nonterminating labelled transition system is an element of a metric space. To apply the unique fixed point proof principle, we introduce a contractive function from this metric space to itself which has the operational model as its fixed point. This function is called a *semantics transformation* and it is also defined by means of a labelled transition system.
B.3.1. Deterministic and nonterminating

We first concentrate on deterministic and nonterminating systems. The operational semantics induced by such a system \( \langle C, A, \rightarrow \rangle \) is an element of the metric space \( C \rightarrow A^\infty \). The corresponding semantics transformation is introduced in

**Definition B.22.** Let \( \langle C, A, \rightarrow \rangle \) be a deterministic and nonterminating labelled transition system. The *semantics transformation* induced by \( \langle C, A, \rightarrow \rangle \) is the function \( \mathcal{T} : [C \rightarrow A^\infty] \) defined by

\[
\mathcal{T}(\mathcal{S})(c) = \begin{cases} 
  e & \text{if } c \xrightarrow{\epsilon}, \\
  a\mathcal{T}(c') & \text{if } c \xrightarrow{a} c'.
\end{cases}
\]

The semantics transformation \( \mathcal{T} \) assigns to the semantics \( \mathcal{S} \) the semantic model \( \mathcal{T}(\mathcal{S}) \). This semantics \( \mathcal{T}(\mathcal{S}) \) maps a terminal configurations to the empty sequence, just like the operational semantics does. To a nonterminal configuration \( c \), the semantics \( \mathcal{T}(\mathcal{S}) \) assigns the sequence \( a\mathcal{T}(c') \), where \( a \) is the label of the transition from \( c \) to some configuration \( c' \) – since the labelled transition system is deterministic this \( c' \) is unique – and the semantics \( \mathcal{S} \) maps this \( c' \) to the sequence \( \mathcal{S}(c') \).

Next, we observe that the semantics transformation has the corresponding operational semantics as its fixed point and that it is a contraction.

**Proposition B.23.** The operational semantics \( \mathcal{O} \) induced by a deterministic and nonterminating labelled transition system is a fixed point of the semantics transformation \( \mathcal{T} \) induced by the labelled transition system, that is,

\[
\mathcal{T}(\mathcal{O}) = \mathcal{O}.
\]

**Proof.** Easy. □

**Proposition B.24.** The semantics transformation induced by a deterministic and nonterminating labelled transition system is contractive.

**Proof.** Let \( \mathcal{T} : [C \rightarrow A^\infty] \) be the semantics transformation induced by the deterministic and nonterminating labelled transition system \( \langle C, A, \rightarrow \rangle \). Let \( \mathcal{S}_1, \mathcal{S}_2 \in C \rightarrow A^\infty \) and \( c \in C \). We will show that

\[
d(\mathcal{T}(\mathcal{S}_1)(c), \mathcal{T}(\mathcal{S}_2)(c)) \leq \frac{1}{2} \cdot d(\mathcal{S}_1, \mathcal{S}_2).
\]

We distinguish two cases.

1. If \( c \xrightarrow{\epsilon} \) then

\[
d(\mathcal{T}(\mathcal{S}_1)(c), \mathcal{T}(\mathcal{S}_2)(c)) = d(\epsilon, \epsilon) = \frac{1}{2} \cdot d(\mathcal{S}_1, \mathcal{S}_2).
\]
(2) If $c \xrightarrow{a} c'$ then
\[
d(T(S_1)(c), T(S_2)(c)) = d(aS_1(c'), aS_2(c')) \leq \frac{1}{2} d(S_1, S_2) \quad \text{[Example A.16(3)].}
\]

Combining the above we arrive at

**Theorem B.25.** The operational semantics $\mathcal{O}$ induced by a deterministic and nonterminating labelled transition system is the unique fixed point of the semantics transformation $\mathcal{T}$ induced by the labelled transition system, that is,
\[
\mathcal{O} = \text{fix}(\mathcal{T}).
\]

**Proof.** Immediate consequence of Propositions B.23, B.24, and Banach’s theorem. □

The above theorem can be exploited as follows. Let $\langle C, A, \rightarrow \rangle$ be a deterministic and nonterminating labelled transition system. If we can show that the semantics $\mathcal{S}: C \rightarrow A^\infty$ is a fixed point of the semantics transformation induced by the system, then we can conclude that the operational semantics induced by the system coincides with the semantics $\mathcal{S}$ by uniqueness of fixed point.

### B.3.2. Nondeterministic and nonterminating

In the rest of this section, we focus on nondeterministic and nonterminating labelled transition systems. The semantics transformations for these systems are defined as follows.

**Definition B.26.** The semantics transformation induced by the labelled transition system $\langle C, A, \rightarrow \rangle$ is the function $\mathcal{T}: [C \rightarrow P_n(A^\infty)]$ defined by
\[
\mathcal{T}(\mathcal{S})(c) = \begin{cases} 
\{\varepsilon\} & \text{if } c \rightarrow, \\
\bigcup \{a\mathcal{S}(c') \mid c \xrightarrow{a} c'\} & \text{otherwise.}
\end{cases}
\]

The semantics $\mathcal{T}(\mathcal{S})$ assigns to a terminal configuration the singleton set consisting of the empty sequence. In the above definition, we denote by $a\mathcal{S}(c')$ the set of sequences $\{a\sigma \mid \sigma \in \mathcal{S}(c')\}$. To a nonterminal configuration $c$, the semantics $\mathcal{T}(\mathcal{S})$ assigns the set of sequences $a\sigma$ obtained from the label $a$ of a transition from the nonterminal configuration $c$ to some configuration $c'$, and a sequence $\sigma$ of $\mathcal{S}(c')$.

**Proposition B.27.** The operational semantics $\mathcal{O}$ induced by a labelled transition system is a fixed point of the semantics transformation $\mathcal{T}$ induced by the labelled transition system, that is,
\[
\mathcal{T}(\mathcal{O}) = \mathcal{O}.
\]
Proof. Let $\mathcal{C}$ and $\mathcal{T}$ be the operational semantics and the semantics transformation induced by the labelled transition system $\langle C, A, \rightarrow \rangle$. Let $c \in C$. Obviously, $\mathcal{T}(\mathcal{C})(c) = \mathcal{C}(c)$ if $c \rightarrow$. Otherwise, for all $\sigma \in \mathcal{P}_n(A^\infty)$,

$$\sigma \in \mathcal{T}(\mathcal{C})(c)$$

$$\iff \exists a \in A : \exists \sigma' \in \mathcal{P}_n(A^\infty) : \exists c' \in C : \sigma = a\sigma' \wedge c \rightarrow c' \wedge \sigma' \in \mathcal{C}(c')$$

$$\iff \sigma \in \mathcal{C}(c). \quad \square$$

According to the above proposition, a semantics transformation has a fixed point. This fixed point is not necessarily unique.

Example B.28. Consider the semantics transformation $\mathcal{T}$ induced by the labelled transition system of Example B.2. According to Proposition B.27, the operational semantics $\mathcal{C}$ of Example B.17 is a fixed point of $\mathcal{T}$. Also the semantics $\mathcal{I}$ defined by

$$\mathcal{I}(c_1) = \{a^n_1 a_2 \mid n \in \mathbb{N}\},$$

$$\mathcal{I}(c_2) = \{e\},$$

is a fixed point of $\mathcal{T}$.

To exploit the semantics transformation $\mathcal{T}$ to relate the operational semantics $\mathcal{C}$ to another semantic model by means of the unique fixed point proof principle, we should turn the semantics transformation into a contractive function from a metric space to itself. For this purpose, we restrict ourselves to the subspace $\mathcal{P}_n(A^\infty)$ of $\mathcal{P}_n(A^\infty)$. Next, we will study semantics transformations which are functions from the metric space $C \rightarrow \mathcal{P}_n(A^\infty)$ to itself. We will see that these semantics transformations are contractive.

We say that a semantics transformation is compactness preserving if it is a function from the space $C \rightarrow \mathcal{P}_n(A^\infty)$ to itself.

Definition B.29. A semantics transformation $\mathcal{T} : [C \rightarrow \mathcal{P}_n(A^\infty)]$ is compactness preserving if $\mathcal{T} \in [C \rightarrow \mathcal{P}_n(A^\infty)]$.

More precisely, the semantics transformation $\mathcal{T}$ is compactness preserving if, for every $\mathcal{I} \in C \rightarrow \mathcal{P}_n(A^\infty)$, we have that $\mathcal{T}(\mathcal{I}) \in C \rightarrow \mathcal{P}_n(A^\infty)$. We will always restrict the compactness preserving semantics transformation $\mathcal{T}$ to the subspace $[C \rightarrow \mathcal{P}_nc(A^\infty)]$.

Exercise B.30. Show that not every labelled transition system induces a compactness preserving semantics transformation.

However, finitely branching labelled transition systems induce compactness preserving semantics transformations.
Theorem B.31. The semantics transformation induced by a finitely branching labelled transition system is compactness preserving.

Proof. Let \( \langle C, A, \rightarrow \rangle \) be a finitely branching labelled transition system. Let \( \mathcal{T} \) be the induced semantics transformation. Let \( \mathcal{S} \in C \rightarrow \mathcal{P}_{\text{nc}}(A^\infty) \) and \( c \in C \). We will show that the set \( \mathcal{T} (\mathcal{S})(c) \) is compact. Obviously, the set \( \mathcal{T} (\mathcal{S})(c) \) is compact if \( c \rightarrow \). Now assume that \( c \rightarrow \). For all \( c' \in C \), the set \( \mathcal{T} (c') \) is compact. Consequently, for all \( c' \in C \) and \( a \in A \), the set \( a\mathcal{T} (c') \) is also compact. Because the labelled transition system is finitely branching, there are only finitely many \( \langle a, c' \rangle \)'s satisfying
\[
 c \xrightarrow{a} c'.
\]
Hence, the set
\[
\{ a\mathcal{T} (c') \mid c \xrightarrow{a} c' \}
\]
is a finite (and hence compact) set of compact sets. According to Michael's theorem, the set \( \mathcal{T} (\mathcal{S})(c) \) is compact. \( \square \)

The operational semantics \( \mathcal{O} \) induced by a finitely branching labelled transition system is compact according to Theorem B.21. Together with Proposition B.27 this gives us that the operational semantics \( \mathcal{O} \) is a fixed point of the compactness preserving semantics transformation \( \mathcal{T} \) induced by the labelled transition system. The uniqueness of this fixed point is derived from the contractiveness of \( \mathcal{T} \).

Proposition B.32. A compactness preserving semantics transformation is contractive.

Proof. Let \( \mathcal{T} : [C \rightarrow \mathcal{P}_{\text{nc}}(A^\infty)] \) be a compactness preserving semantics transformation. Let \( \mathcal{S}_1, \mathcal{S}_2 \in C \rightarrow \mathcal{P}_{\text{nc}}(A^\infty) \) and \( c \in C \). We will show that
\[
d(\mathcal{T} (\mathcal{S}_1)(c), \mathcal{T} (\mathcal{S}_2)(c)) \leq \frac{1}{2} \cdot d(\mathcal{S}_1, \mathcal{S}_2).
\]
We distinguish two cases.

- If \( c \rightarrow \), then
  \[
d(\mathcal{T} (\mathcal{S}_1)(c), \mathcal{T} (\mathcal{S}_2)(c))
  = d(\{c\}, \{c\})
  \leq \frac{1}{2} \cdot d(\mathcal{S}_1, \mathcal{S}_2).
\]

- Let \( c \rightarrow \). Assume \( a\sigma_1 \in \mathcal{T} (\mathcal{S}_1)(c) \). Then there exists a \( c' \in C \) such that \( c \xrightarrow{a} c' \) and \( \sigma'_1 \in \mathcal{S}_1(c') \). Because there exists a \( \sigma'_2 \in \mathcal{S}_2(c') \) such that
  \[
d(\sigma'_1, \sigma'_2)
  \leq d(\mathcal{S}_1(c'), \mathcal{S}_2(c'))
  \leq d(\mathcal{S}_1, \mathcal{S}_2),
\]
we have that $a/\sigma_2 \in T(S_2)(c)$ and
\[
d(a/\sigma_1', a/\sigma_2') = \frac{1}{2} \cdot d(\sigma_1', \sigma_2') \quad \text{[Example A.16(3)]}
\]
\[\leq \frac{1}{2} \cdot d(S_1, S_2). \quad \square
\]

**Exercise B.33.** Show that the fact that the semantic models considered assign to each configuration a *nonempty* set is essential in the above proof.

From Banach’s theorem we can conclude that $\mathcal{O}$ is the unique fixed point of $\mathcal{T}$.

**Theorem B.34.** The operational semantics $\mathcal{O}$ induced by a finitely branching labelled transition system is the unique fixed point of the semantics transformation $\mathcal{T}$ induced by the labelled transition system, that is,
\[
\mathcal{O} = \text{fix}(\mathcal{T}).
\]

**Proof.** Immediate consequence of Propositions B.27 and B.32 and Banach’s theorem. \(\square\)

**Appendix C. Metric labelled transition systems**

In Section B.2.3, we studied operational semantic models defined by means of nondeterministic and nonterminating labelled transition systems. We focussed on finitely branching labelled transition systems. For these systems we developed some theory to prove the induced operational semantic models equal to other semantic models by uniqueness of fixed point in Sections B.2.3 and B.3.2.

A large variety of nondeterministic languages can be modelled operationally by means of a finitely branching labelled transition system. However, there are languages which cannot be captured by a labelled transition system satisfying this finiteness condition. An example is the language considered in Section 5. In that section, we study a timed specification language.

In this appendix, we *generalise* the results of Sections B.2.3 and B.3.2. This will allow us to deal with a considerably larger class of languages, including the timed language of Section 5. Our generalisation is based on the fact that in (metric) topology finiteness is a special case of *compactness*. Every finite subset of a metric space is compact and for every compact subset of a metric space we can find a finite subset which is arbitrary close to it. To exploit this fact, we supply the labelled transition systems with some additional metric structure. This structure is added by endowing the set of configurations and the set of actions both with a complete metric. These enriched labelled transition systems we call *metric labelled transition systems*. The additional metric structure enables us to generalise the finiteness condition finitely branching. We generalise from finitely branching to *compactly branching* and *nonexpansive*. A metric
A metric labelled transition system is compactly branching if every configuration has a compact set of outgoing transitions, and it is nonexpansive if transitioning is nonexpansive. For metric labelled transition systems satisfying this generalised finiteness condition we extend the results of Sections B.2.3 and B.3.2.

In Section C.1, we introduce metric labelled transition systems and the generalised finiteness condition compactly branching and nonexpansive. Operational semantic models induced by metric labelled transition systems are studied in Section C.2. It is proved that a compactly branching and nonexpansive metric labelled transition system induces a compact and nonexpansive operational semantics. In Section C.3, we focus on semantics transformations for metric labelled transition systems. We show that a compactly branching and nonexpansive metric system defines a compactness and nonexpansiveness preserving semantics transformation, and that the operational semantics induced by the system is the unique fixed point of this transformation. Like in Appendix B, we keep the examples in this appendix as simple as possible. For more about metric labelled transition systems we refer the reader to [19] on which this appendix is based.

C.1. Metric labelled transition systems

In this section, we introduce the notion of a metric labelled transition system. A metric labelled transition system is a labelled transition system with some additional structure. That is, the set of configurations and the set of actions are both endowed with a complete metric.

Definition C.1. A metric labelled transition system is a triple \( \langle C, A, \rightarrow \rangle \) consisting of
- a complete metric space of configurations \( C \),
- a complete metric space of actions \( A \), and
- a transition relation \( \rightarrow \subseteq C \times A \times C \).

Example C.2. The labelled transition system

\[
\begin{align*}
0 & \xrightarrow{a} \frac{1}{2} \quad \text{for } a \in [0, 1] \\
0 & \xrightarrow{a} 1 \quad \text{for } a \in [0, 1] \\
1 & \xrightarrow{a} 1
\end{align*}
\]

depicted by

![Diagram](image)

can be turned into a metric labelled transition system by endowing the set of configurations \( \{0, \frac{1}{2}, 1\} \) and the set of actions \([0, 1]\) both with the Euclidean metric.
Because we have a metric on the sets of configurations and actions (and hence on the Cartesian product of these sets), the finiteness condition finitely branching can be generalised to compactly branching: for each configuration, its set of outgoing transitions is compact.

**Definition C.3.** A metric labelled transition system \( \langle C, A, \rightarrow \rangle \) is **compactly branching** if, for all \( c \in C \), the set
\[
\mathcal{S}(c) = \{ \langle a, c' \rangle \mid c \xrightarrow{a} c' \}
\]
is compact.

If we endow the configurations and the actions of a finitely branching labelled transition system both with an arbitrary complete metric, then we obtain a compactly branching metric labelled transition system. A compactly branching metric labelled transition system is in general not finitely branching.

**Example C.4.** The metric labelled transition system introduced in Example C.2 is not finitely branching but it is compactly branching. If, in this example, we endow the actions with the discrete metric, the metric labelled transition system so obtained is no longer compactly branching.

For a compactly branching metric labelled transition system we introduce the condition of transitioning being nonexpansive. To formulate this condition we provide the compact sets of outgoing transitions of the configurations, elements of \( \mathcal{P}_c(A \times C) \), with a metric. The set of action–configuration pairs is endowed with the metric obtained from the metric on the actions and the metric on the configurations multiplied by a \( \frac{1}{2} \), and the resulting space is denoted by \( A \times \frac{1}{2} \cdot C \) (see Definition A.5(1) and D.1). As we will see below, the introduction of the \( \frac{1}{2} \) gives rise to a less restrictive condition. The compact sets of these pairs are endowed with the Hausdorff metric (see Definition A.19 and Eq. (A.9)).

**Definition C.5.** A compactly branching metric labelled transition system \( \langle C, A, \rightarrow \rangle \) is **nonexpansive** if the function \( \mathcal{S} : C \rightarrow \mathcal{P}_c(A \times \frac{1}{2} \cdot C) \) defined by
\[
\mathcal{S}(c) = \{ \langle a, c' \rangle \mid c \xrightarrow{a} c' \}
\]
is nonexpansive.

**Example C.6.** The metric labelled transition system of Example C.2 is not nonexpansive, because
\[
d(\mathcal{S}(\frac{1}{2}), \mathcal{S}(1)) = d(\emptyset, \{\{1,1\}\})
\]
By adding the transition
\[
\frac{1}{2} \rightarrow \frac{1}{2}
\]
we obtain the compactly branching metric labelled transition system

which is nonexpansive.

The \(\frac{1}{2}\) in the above definition does not change the compactness condition. By leaving out the \(\frac{1}{2}\) in the above definition, we obtain a more restrictive condition.

**Example C.7.** The metric labelled transition system

\[
\begin{align*}
\frac{1}{4} & \rightarrow 0 \\
\frac{3}{4} & \rightarrow 1
\end{align*}
\]

depicted by

with the set of configurations \(\{0, \frac{1}{4}, \frac{3}{4}, 1\}\) endowed with the Euclidean metric, is nonexpansive, since

\[
d(\mathcal{F}(\frac{1}{4}), \mathcal{F}(\frac{3}{4})) \\
= d(\{(0,0)\}, \{(0,1)\}) \\
= \frac{1}{4} \\
\leq \frac{1}{3} \\
= d(\frac{1}{4}, \frac{3}{4}).
\]
If we leave out the $\frac{1}{2} \cdot$ we have that
\[
\begin{align*}
    d(\mathcal{I}(\frac{1}{4}), \mathcal{I}(\frac{3}{4})) &= d(\{\langle 0,0 \rangle \}, \{\langle 0,1 \rangle \}) \\
    &= \frac{1}{2} \\
    &\geq \frac{1}{3} \\
    &= d(\frac{1}{4}, \frac{3}{4}).
\end{align*}
\]

A finitely branching labelled transition system with the configurations endowed with the discrete metric and the actions endowed with an arbitrary complete metric is (compactly branching and) nonexpansive. Consequently, we have generalised from finitely branching to compactly branching and nonexpansive.

**Proposition C.8.** A labelled transition system is finitely branching if and only if the metric labelled transition system obtained by endowing the configurations and actions with the discrete metric is compactly branching and nonexpansive.

**Proof.** Trivial. □

**C.2. Operational semantics**

Like in Section B.2.3, we consider operational semantic models induced by nondeterministic and nonterminating systems. In this section, we concentrate on metric labelled transition systems. Again, the operational semantics assigns to each configuration of the system a nonempty set of finite and infinite action sequences. These action sequences and their metric are introduced in

**Definition C.9.** Let $A$ be a complete metric space. The complete metric space $(\sigma \in) A^\infty$ is defined as the solution of the recursive equation
\[
A^\infty \cong \{e\} + (A \times \frac{1}{2} \cdot A^\infty).
\]

According to Theorem D.3, the above recursive equation has a unique solution. As we mention in Exercise D.4, if we take $A$ to be a set endowed with the discrete metric, then the set of finite and infinite sequences over $A$ endowed with the Baire metric is the solution of the above equation.

**Exercise C.10.** The set underlying the metric space $A^\infty$ is the set of finite and infinite sequences over the set underlying the metric space $A$. Specify the metric of the space $A^\infty$ in terms of sequences (like the Baire metric).

The operational semantics induced by a metric labelled transition system is defined as in Definition B.16.
**Definition C.11.** The operational semantics induced by the metric labelled transition system $\langle C, A, \rightarrow \rangle$ is the function $\mathcal{O} : C \rightarrow \mathcal{P}(A^\omega)$ defined by

$$\mathcal{O}(c) = \{a_1a_2 \cdots a_n \mid c = c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} c_n \} \cup \{a_1a_2 \cdots \mid c = c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} \cdots \}.$$ 

The only difference with Definition B.16 is that the set $A^\omega$, and hence the set $\mathcal{P}(A^\omega)$, is endowed with different metric (this difference is not visible in the above definition).

**Example C.12.** The metric labelled transition system of Example C.2 induces the operational semantics $\mathcal{O}$ defined by

$$\mathcal{O}(0) = [0, 1] \cdot \{^\omega\} \cup [0, 1],$$

$$\mathcal{O}(\frac{1}{2}) = \{^\omega\},$$

$$\mathcal{O}(1) = \{^\omega\}.$$ 

To apply the unique fixed point proof principle, we want the operational semantics to be an element of a metric space. As before, we restrict ourselves to the subspace $\mathcal{P}(A^\omega)$ of $\mathcal{P}(A^\omega)$.

Like in Definition B.18, we introduce the compactness of an operational semantics in

**Definition C.13.** An operational semantics $\mathcal{O} : C \rightarrow \mathcal{P}(A^\omega)$ is compact if $\mathcal{O} \in C \rightarrow \mathcal{P}(A^\omega)$.

**Example C.14.** The operational semantics presented in Example C.12 is compact if the action set $[0, 1]$ is endowed with the Euclidean metric. If we endow the action set $[0, 1]$ with the discrete metric then the operational semantics is not compact any more.

Because compact operational semantic models are element of a metric space, we can possibly exploit the unique fixed point proof principle to relate these operational semantic models to other semantic models.

As the above example shows us, not every metric labelled transition system induces a compact operational semantics. If we restrict ourselves to compactly branching and nonexpansive metric labelled transition systems, then we obtain compact operational semantic models. Without the additional nonexpansiveness condition we do in general not obtain compact operational semantic models.

**Exercise C.15.** Give an example of a compactly branching metric labelled transition system which does not give rise to a compact operational semantics.

Note that a finitely branching metric labelled transition system induces a compact operational semantics (the proof of Theorem B.21 does not depend on the metrics the configurations and actions are endowed with).
Next, we prove that a compactly branching and nonexpansive metric labelled transition system induces a compact operational semantics. To verify this result we prove two additional propositions. In the first proposition, we demonstrate that the nonterminal and terminal configurations of a compactly branching and nonexpansive metric labelled transition system are distance 1 apart.

**Proposition C.16.** The nonterminal and terminal configurations of a compactly branching and nonexpansive metric labelled transition system are distance 1 apart.

**Proof.** Left to the reader as an exercise. □

**Exercise C.17.** Prove Proposition C.16.

In the second proposition we show that, for a compactly branching and nonexpansive metric labelled transition system, for all configurations \( c \) and natural numbers \( n \), the set of transition sequences starting from the configuration \( c \) and truncated at length \( n \) is compact.

**Proposition C.18.** Let \( \langle C,A,\rightarrow \rangle \) be a compactly branching and nonexpansive metric labelled transition system. For all \( c \in C \) and \( n \in \mathbb{N} \), the set

\[
S_n(c) = \{ \langle a_1,c_1,\ldots,a_n,c_n \rangle \mid c = c_0 \xrightarrow{a_1} c_1 \cdots \xrightarrow{a_n} c_n \}
\]

is compact.

**Proof.** This proposition is proved by induction on \( n \). For \( n = 0 \) the proposition is vacuously true. Let \( n > 0 \). Let \( c \in C \). Because the metric labelled transition system is compactly branching, for all \( c_{n-1} \in C \), the set \( S(c_{n-1}) \) is compact. Consequently, for all \( c_1,\ldots,c_{n-1} \in C \) and \( a_1,\ldots,a_{n-1} \in A \), the set

\[
\{ \langle a_1,c_1,\ldots,a_n,c_n \rangle \mid \langle a_n,c_n \rangle \in S(c_{n-1}) \}
\]

is also compact. Since the metric labelled transition system is nonexpansive, the function assigning to \( \langle a_1,c_1,\ldots,a_{n-1},c_{n-1} \rangle \) the above set is nonexpansive. By induction, the set \( S^{n-1}(c) \) is compact. From Alexandroff’s theorem we can conclude that

\[
\{ \langle a_1,c_1,\ldots,a_n,c_n \rangle \mid \langle a_n,c_n \rangle \in S(c_{n-1}) \} \cup \{ \langle a_1,c_1,\ldots,a_{n-1},c_{n-1} \rangle \} \in S^{n-1}(c)
\]

is a compact set of compact sets. It follows from Michael’s theorem that the set

\[
\bigcup \{ \langle a_1,c_1,\ldots,a_n,c_n \rangle \mid \langle a_n,c_n \rangle \in S(c_{n-1}) \} \cup \{ \langle a_1,c_1,\ldots,a_{n-1},c_{n-1} \rangle \} \in S^{n-1}(c),
\]

that is, \( S^n(c) \), is compact. □

Now we are ready to prove
Theorem C.19. The operational semantics induced by a compactly branching and nonexpansive metric labelled transition system is compact.

Proof. Let \( \langle C, A, \rightarrow \rangle \) be a compactly branching and nonexpansive metric labelled transition system. We prove that the induced operational semantics \( \mathcal{O} \) is compact, that is, for all \( c \in C \), the set \( \mathcal{O}(c) \) is compact.

Let \( c \in C \). Let \( (\sigma_n)_n \) be a sequence in \( \mathcal{O}(c) \). We show that there exists a subsequence \( (\sigma_{n(n)})_n \) of \( (\sigma_n)_n \) converging to some \( \sigma \in \mathcal{O}(c) \).

The subsequence \( (\sigma_{n(n)})_n \) will be constructed from a collection of subsequences \( (\sigma_{n(n)})_n \) satisfying

\[
(\forall m \in \mathbb{N}: Q(m)) \lor (\exists k \in \mathbb{N}: \forall 1 \leq m < k: Q(m) \land R(k)) \tag{C.11}
\]

where

\[
Q(m) \iff \forall n \in \mathbb{N}: \sigma_{n(n)} = a_{1,n(n)} a_{2,n(n)} \cdots a_{m,n(n)} \sigma_{m,n(n)} \land \\
c = c_0 a_{1,n(n)} c_{1,n(n)} a_{2,n(n)} \cdots a_{m,n(n)} c_{m,n(n)} \rightarrow \land \\
\sigma_{m,n(n)} \in \mathcal{O}(c(n(n))) \land \\
\forall 1 \leq j \leq m: \lim_{n} a_{j,n(n)} = a_j \land \lim_{n} c_{j,n(n)} = c_j \land \\
c = c_0 a_1 c_1 \cdots c_n \rightarrow .
\]

and

\[
R(m) \iff \forall n \in \mathbb{N}: \sigma_{n(n)} = a_{1,n(n)} a_{2,n(n)} \cdots a_{m,n(n)} \land \\
c = c_0 a_{1,n(n)} c_{1,n(n)} a_{2,n(n)} \cdots a_{m,n(n)} c_{m,n(n)} \rightarrow \land \\
\forall 1 \leq j \leq m: \lim_{n} a_{j,n(n)} = a_j \land \lim_{n} c_{j,n(n)} = c_j \land \\
c = c_0 a_1 c_1 \cdots c_n \rightarrow .
\]

The existence of the subsequences \( (\sigma_{n(n)})_n \) is verified by proving that, for all \( i \in \mathbb{N} \),

\[
P(i) \iff (\forall 1 \leq m \leq i: Q(m)) \lor (\exists 1 \leq k \leq i: \forall 1 \leq m < k: Q(m) \land R(k))
\]

by induction on \( i \).

To prove \( P(0) \) it suffices to show \( Q(0) \lor R(0) \). Obviously, the sequence \( (\sigma_n)_n \) satisfies \( Q(0) \lor R(0) \).

Let \( i > 0 \). To prove \( P(i-1) \Rightarrow P(i) \) it suffices to show \( Q(i-1) \Rightarrow Q(i) \lor R(i) \). If \( Q(i-1) \), then

\[
\forall n \in \mathbb{N}: ((\sigma_{i-1,n(n)}) = a_{1,i-1,n(n)} a_{2,i-1,n(n)} \cdots a_{i,i-1,n(n)} \sigma_{i,i-1,n(n)} \land \\
c = c_0 a_{1,i-1,n(n)} c_{1,i-1,n(n)} a_{2,i-1,n(n)} \cdots a_{i,i-1,n(n)} c_{i,i-1,n(n)} \rightarrow \land
\]
Given a finitely branching labelled transition system \( \langle C, A, \to \rangle \), we endow the action set \( A \) with the discrete metric (consequently, the metric on \( A^\infty \) becomes the Baire metric) and the configuration set \( C \) also with the discrete metric. According to
Proposition C.8 we obtain a compactly branching and nonexpansive metric labelled transition system. By Theorem C.19 the corresponding operational semantics is compact. □

The operational semantics induced by a compactly branching and nonexpansive metric labelled transition system has another property besides being compact: it is nonexpansive. The nonexpansiveness of a compact operational semantics is exploited when we want to apply the unique fixed point proof principle (the details will be supplied in Section C.3).

**Theorem C.21.** The compact operational semantics induced by a compactly branching and nonexpansive metric labelled transition system is nonexpansive.

**Proof.** Let $⟨C,A,→⟩$ be a compactly branching and nonexpansive metric labelled transition system. Let $O$ be the induced compact operational semantics. To prove the non-expansiveness of $O$, a sequence $(O_n)_n$ of nonexpansive functions converging to $O$ is introduced. Because the space of nonexpansive functions $C \rightarrow \mathcal{P}_{nc}(A^\infty)$ is complete (Proposition A.30), we can conclude that $O$ is nonexpansive. The function $O_n : C \rightarrow \mathcal{P}_{nc}(A^\infty)$ is defined by

$$
O_n(c) = \begin{cases} 
\{ \varepsilon \} & \text{if } c \not\rightarrow, \\
\bigcup \{ aO_{n-1}(c') \ | c \xrightarrow{a} c' \} & \text{otherwise}. 
\end{cases}
$$

We have left to prove that, for all $n$, $O_n \in C \rightarrow \mathcal{P}_{nc}(A^\infty)$. We prove this by induction on $n$. Obviously, $O_0 \in C \rightarrow \mathcal{P}_{nc}(A^\infty)$. Assume $n > 0$. Let $c \in C$. By definition,

$$
O_n(c) = \begin{cases} 
\{ \varepsilon \} & \text{if } c \not\rightarrow, \\
\bigcup \{ aO_{n-1}(c') \ | c \xrightarrow{a} c' \} & \text{otherwise}. 
\end{cases}
$$

Clearly, the set $O_n(c)$ is nonempty.

Next, we show that the set $O_n(c)$ is compact. Because the metric labelled transition system is compactly branching, the set $O(c)$ is compact. By induction, $O_{n-1}$ delivers compact sets. One can easily verify that, for all $a \in A$ and $c' \in C$, the set $aO_{n-1}(c')$ is compact. Since $O_{n-1}$ and action prefixing are nonexpansive (induction and Example A.16(3)), and the nonexpansive image of a compact set is compact (Alexandroff’s theorem),

$$
\{ aO_{n-1}(c') \ | c \xrightarrow{a} c' \}
$$

is a compact set of compact sets. According to Michael’s theorem, the set

$$
\bigcup \{ aO_{n-1}(c') \ | c \xrightarrow{a} c' \}
$$

is compact. Also $\{ \varepsilon \}$ is a compact set. Hence, the set $O_n(c)$ is compact.

Finally, the nonexpansiveness of $\mathcal{C}_n$ is proved. We have to show that, for all $c_1, c_2 \in C$,
\[
d(\mathcal{C}_n(c_1), \mathcal{C}_n(c_2)) \leq d(c_1, c_2).
\]
If both $c_1$ and $c_2$ are terminal configurations then the above is vacuously true. Because the nonterminal and terminal configurations are distance 1 apart (Proposition C.16), the above is also true if one of the configurations is a nonterminal configuration and the other one is a terminal configuration. That leaves us only the case that both $c_1$ and $c_2$ are nonterminal configurations. Let $a_1 \sigma_1 \in \mathcal{C}_n(c_1)$. Then there exists a $c'_1 \in C$ such that $c_1 \xrightarrow{a_1} c'_1$ and $\sigma_1 \in \mathcal{C}_{n-1}(c'_1)$. Since the metric labelled transition system is nonexpansive, $c_2 \xrightarrow{a_2} c'_2$ for some $a_2 \in A$ and $c'_2 \in C$ such that $d(a_1, a_2) \leq d(c_1, c_2)$ and $d(c'_1, c'_2) \leq 2 \cdot d(c_1, c_2)$. Hence, there exists a $\sigma_2 \in \mathcal{C}_{n-1}(c'_2)$ such that
\[
d(\sigma_1, \sigma_2)
\leq d(\mathcal{C}_{n-1}(c'_1), \mathcal{C}_{n-1}(c'_2))
\leq d(c'_1, c'_2) \quad \text{[by induction $\mathcal{C}_{n-1}$ is nonexpansive]}
\leq 2 \cdot d(c_1, c_2).
\]
Consequently, $a_2 \sigma_2 \in \mathcal{C}_n(c_2)$ and
\[
d(a_1 \sigma_1, a_2 \sigma_2)
= \max\{d(a_1, a_2), \frac{1}{2} \cdot d(\sigma_1, \sigma_2)\}
\leq d(c_1, c_2). \quad \square
\]

**Exercise C.22.** Prove that for all $n \in \mathbb{N}$ and $c \in C$, $d(\mathcal{C}_n(c), \mathcal{C}(c)) \leq 2^{-n}$.

### C.3. Semantics transformations

Next, we introduce semantics transformations for (nondeterministic and nonterminating) metric labelled transition systems. Similar to Definition B.26, we define the semantics transformation induced by a metric labelled transition system in

**Definition C.23.** The **semantics transformation** induced by the metric labelled transition system $\langle C, A, \rightarrow \rangle$ is the function $\mathcal{F} : [C \rightarrow \mathcal{P}(A^\infty)]$ defined by

\[
\mathcal{F}(\mathcal{S})(c) = \begin{cases} 
\{c\} & \text{if } c \not\xrightarrow{\cdot} \\
\bigcup \{a \mathcal{F}(c') \mid c \xrightarrow{a} c'\} & \text{otherwise}.
\end{cases}
\]

The semantics transformation and the operational semantics induced by the same metric labelled transition system are related as in Proposition B.27.

**Proposition C.24.** The operational semantics $\mathcal{C}$ induced by a metric labelled transition system is a fixed point of the semantics transformation $\mathcal{F}$ induced by the metric
labelled transition system, that is,
\[ \mathcal{T}(\emptyset) = \emptyset. \]

**Proof.** Similar to the proof of Proposition B.27. \( \square \)

As we have already seen in Example B.28, a semantics transformation does not have a unique fixed point in general. However, a semantics transformation which is a contractive function from a metric space to itself does. Therefore, we consider semantics transformations transforming compact (and nonexpansive) semantic models.

**Definition C.25.** A semantics transformation \( \mathcal{T} : [C \rightarrow \mathcal{P}(A^\omega)] \) is **compactness preserving** if \( \mathcal{T} \in [C \rightarrow \mathcal{P}_{nc}(A^\omega)] \).

Not every compactly branching and nonexpansive metric labelled transition system induces a compactness preserving semantics transformation.

**Example C.26.** The metric labelled transition system
\[ c \xrightarrow{a} c' \quad \text{for } c, c' \in [0, 1] \text{ and } a \in [0, 1] \]
with the configurations and the actions endowed with the Euclidean metric, is compactly branching and nonexpansive. Given the compact semantics \( \mathcal{S} \) defined by
\[ \mathcal{S}(c) = \begin{cases} \{1^n\} & \text{if } c = \frac{1}{n} \text{ for some } n > 0, \\ \{a\} & \text{otherwise}, \end{cases} \]
the semantics \( \mathcal{T}(\mathcal{S}) \) is not compact, since the set
\[ \mathcal{T}(\mathcal{S})(0) = [0, 1] \cup [0, 1] \cdot \{1^n | n > 0\} \]
is not compact.

We restrict ourselves to nonexpansive and compact semantics.

**Definition C.27.** A compactness preserving semantics transformation \( \mathcal{T} : [C \rightarrow \mathcal{P}_{nc}(A^\omega)] \) is **nonexpansiveness preserving** if \( \mathcal{T} \in [C \rightarrow \mathcal{P}_{nc}(A^\omega)] \).

Not every metric labelled transition system induces a compactness and nonexpansiveness preserving semantics transformation.

**Example C.28.** Consider the metric labelled transition system
\[ 0 \xrightarrow{\frac{1}{n}} \frac{1}{n}, \text{ for } n > 0 \]
with the configurations and the actions endowed with the Euclidean metric. Although
the semantics $\mathcal{S}$, defined by, for all $c$,
$$\mathcal{S}(c) = \{e\}$$
is compact, the semantics $\mathcal{T}(\mathcal{S})$ is not compact, since the set
$$\mathcal{T}(\mathcal{S})(0) = \{\frac{1}{n} \mid n > 0\}$$
is not compact.

Not even a compactly branching metric labelled transition system necessarily induces
a compactness and nonexpansiveness preserving semantics transformation.

**Example C.29.** If we add to the metric labelled transition system of Example C.28
the transition
$$0 \xrightarrow{0} 0$$
then we obtain a compactly branching system. The semantics $\mathcal{S}$, defined by, for all $c$,
$$\mathcal{S}(c) = \{e\}$$
is compact and nonexpansive. The semantics $\mathcal{T}(\mathcal{S})$ is compact but not nonexpansive.

But a compactly branching and nonexpansive metric labelled transition system gives
rise to a compactness and nonexpansiveness preserving semantics transformation.

**Theorem C.30.** The semantics transformation induced by a compactly branching and
nonexpansive metric labelled transition system is compactness and nonexpansiveness
preserving.

**Proof.** Similar to the induction step of the proof of Theorem C.21.

As a consequence of the above theorem we have Theorem B.31.

**Corollary C.31.** The semantics transformation induced by a finitely branching labelled
transition system is compactness preserving.

**Proof.** Given a finitely branching labelled transition system $\langle C, A, \rightarrow \rangle$, we endow the
configuration set $C$ and the action set $A$ with the discrete metric. By Proposition C.8,
we obtain a compactly branching and nonexpansive metric labelled transition system.
According to Theorem C.30, the corresponding semantics transformation is compactness
and nonexpansiveness preserving. Because the configuration set $C$ is endowed with the
discrete metric, every function from $C$ to $\mathcal{P}(A^\infty)$ is nonexpansive. Consequently, the
induced semantics transformation is compactness preserving.
A compactness and nonexpansiveness preserving semantics transformation is a function from a metric space to itself. According to Proposition C.24 and Theorems C.19 and C.21, the corresponding operational semantics is a fixed point of the semantics transformation. Note that the nonexpansiveness plays a crucial role here (see Example C.26). To be able to apply the unique fixed point proof principle we have left to prove that the semantics transformation is a contraction.

**Proposition C.32.** A compactness and nonexpansiveness preserving semantics transformation is contractive.

**Proof.** Similar to the proof of Proposition B.32. □

Combining the above results, we arrive at

**Theorem C.33.** The operational semantics $\mathcal{O}$ induced by a compactly branching and nonexpansive metric labelled transition system is the unique fixed point of the semantics transformation $\mathcal{F}$ induced by the metric labelled transition system, that is,

$$\mathcal{O} = \text{fix}(\mathcal{F}).$$

Theorem B.34 is a corollary of the above theorem.

**Appendix D. Recursive equations**

Almost all our denotational semantic models use complete metric spaces. Most of these spaces are defined as the solution of a recursive equation. In this appendix, we discuss how to build these equations. Furthermore, we specify a class of equations which has unique solutions.

In the equations, we encounter various operations on metric spaces. Well-known operations on sets, like the Cartesian product $\times$, the disjoint union $+$, the function space $\to$, and the powerset $\mathcal{P}$, can be lifted to operations on metric spaces as we have seen in Section A.1 and A.3. We restrict $\to$ to the nonexpansive function space $\mathcal{P}_{\text{nc}}$, and $\mathcal{P}$ to the nonempty and compact hyperspace $\mathcal{P}_{\text{nc}}$. Furthermore, the operation $\frac{1}{2} \cdot$ on metric spaces is introduced below. Applied to a metric space, this operation leaves the set unchanged and multiplies the metric by a $\frac{1}{2}$. This operation plays an important role because it appears in all the equations we will consider and since it is crucial for the uniqueness result presented below.

In this paper, we encounter recursive equations like

$$X \cong \{e\} + (A \times \frac{1}{2} \cdot X), \quad (D.1)$$

$$Y \cong A \rightarrow_{\frac{1}{2}} \mathcal{P}_{\text{nc}}(A \times \frac{1}{2} \cdot Y). \quad (D.2)$$
In this appendix, we will not discuss the semantic considerations leading to these equations. The equations are all of the form

\[ X \cong E \quad \text{(D.3)} \]

and define a complete metric space \( X \) which is isometric – this is denoted by \( \cong \) and is defined below – to \( E \). This \( E \) is built from

- the above-mentioned operations on metric spaces,
- some given complete metric spaces – usually sets endowed with the discrete metric – and
- the complete metric space \( X \).

The occurrence of \( X \) in \( E \) makes the equation recursive. We say that the defined metric space \( X \) is the solution of the equation. In general, a recursive equation might have no solution or several solutions. Below, we identify a class of equations which have unique (up to isometry) solutions. This is done by specifying a grammar for \( E \). This grammar does not give us all the equations that have unique solutions, but it covers the equations we want to solve. For example, it handles Eqs. (D.1) and (D.2).

Although a solution of Eq. (D.3) is a complete metric space \( X \) being isometric to \( E \), we will not write the isometries when going from \( X \) to \( E \) or vice versa. They can be put in without any difficulty, but will clutter up the presentation.

In Section D.1, we introduce the remaining ingredients we need to build equations. A class of equations with unique solutions is singled out in Section D.2. In this appendix, we do not strive for a very general theory of solving recursive equations. We present a simple one which serves our purposes. We also refrain from giving proofs. Those can be found in [4].

### D.1. Building equations

Most ingredients of the recursive equations have already been discussed. We have left the introduction the operation \( \frac{1}{2} \cdot \) and the equivalence \( \cong \).

We start with the definition of the operation \( \frac{1}{2} \cdot \):

\textbf{Definition D.1.} Let \( X \) be a metric space. The metric \( (\frac{1}{2} \cdot d)_X : X \times X \to [0, 1] \) is defined by

\[ (\frac{1}{2} \cdot d)_X(x, y) = \frac{1}{2} \cdot d_X(x, y). \]

Note that this operation preserves completeness. Next, we introduce the natural notion of equivalence on metric spaces.

\textbf{Definition D.2.} The metric spaces \( X \) and \( Y \) are isometric if there exists a bijective function \( f : X \to Y \) satisfying for all \( x_1, x_2 \in X \),

\[ d_Y(f(x_1), f(x_2)) = d_X(x_1, x_2). \]
D.2. Solving equations

As we already mentioned above, not every equation has a solution (we are not aware of a simple example illustrating this fact). An equation might also have several solutions. Obviously,

\[ X \cong X \]

has more than one. In the next theorem, we identify a class of equations which all have a unique (up to isometry) solution. This class does not contain all the equations which have a unique solution. For example, the equation

\[ X \cong A + \frac{1}{2} \cdot (\frac{1}{2} \cdot X \rightarrow \frac{1}{2} \cdot X) \]

has a unique solution, but is not dealt with in the next theorem. However, the theorem handles all the equations we encounter in the rest of this paper.

Theorem D.3. Let

\[ E ::= A | \frac{1}{2} \cdot X | \mathcal{P}_{nc}(E) | E \times E | E + E | A \rightarrow_1 E \]

where \( A \) can be any complete metric space. The equation \( X \cong E \) has a unique (up to isometry) solution.

As the reader can easily verify, the above theorem applies to Eqs. (D.1) and (D.2) presented above. Consequently, these equations have a unique solution.

Exercise D.4. Prove that the set \( A^\infty \) of finite and infinite sequences over the set \( A \) endowed with the Baire metric is the solution of Eq. (D.1) where the set \( A \) is provided with the discrete metric.

References


