stichting
mathematisch
centrum

MC

AFDELING INFORMATICA
(DEPARTMENT OF COMPUTER SCIENCE)

IW 169/81 JULI

R. KUIPER

AN OPERATIONAL SEMANTICS FOR BOUNDED NONDETERMINISM EQUIVALENT TO A DENOTATIONAL ONE

Preprint

kruislaan 413 1098 SJ amsterdam

Printed at the Mathematical Centre, 413 Kruislaan, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

1980 Mathematics subject classification: 6810

ACM-Computing Reviews Categories: 5.24, 4.22

An operational semantics for bounded nondeterminism equivalent to a denotational one *)

by

R. Kuiper

ABSTRACT

Dyadic nondeterministic choice is added to the programming language with recursive procedures as used in de Bakker's monograph on program correctness [5]. This leads to considerable changes in the operational semantics. The possible result of the execution of a program is no more given as a single state, but as a set of possible states. Furthermore, the execution of a program is no more given as a computation sequence but as a set of possible computation sequences with tree-like properties.

We present a "natural" operational semantics O defined by means of a function *Comp*, where *Comp* yields for each program R and each state σ a set of computation sequences, characterized by equations in the style of COOK [7]. For this set of equations we prove, in a topological setting, the existence of a unique solution and the equivalence of the operational semantics to the usual denotational one, defined by fixed point techniques.

KEY WORDS & PHRASES: *Operational semantics, denotational semantics, bounded nondeterminism*

---

*) This report will be submitted for publication elsewhere.

0. INTRODUCTION

The subject of this paper is to investigate the effects of adding bounded nondeterministic choice to a simple language with recursive procedures on the definition and properties of the operational semantics

The motivation to introduce an operational semantics is the following usual one. A method for proving program correctness is to abstract to a more mathematical level by defining a denotational semantics and to give a proof system on that level. A way to justify this abstraction is to define an operational semantics such that on the one hand it is intuitively close to the actual program execution and on the other hand can be proved to be equivalent to the denotational semantics. We provide a "natural" operational semantics; its justification and the proof of its equivalence to a denotational one are the main aims of this paper.

The reasons to add dyadic nondeterministic choice - as will be seen later, extension to finite choice introduces no extra problems - are twofold. Firstly, in practice nondeterministic choice enters the scene directly, cf. DIJKSTRA's guarded command [9], as well as indirectly, cf. parallellism and concurrency [12], where one process is selected to proceed, or one communication is selected to be executed. Secondly, in theory nondeterministic choice is a fairly easy setting in which tree-like structures appear instead of computation sequences as when dealing with deterministic sequential programs. This phenomenon also occurs as soon as parallel programming and concurrent processes are concerned and introduces considerable changes in the theoretical treatment. Contrary to the deterministic case, justification of the defined operational semantics in view of existence and uniqueness of the described function is not a clear case, and thus grew into a next-important aim in itself.

The framework we use is that developed in DE BAKKER's monograph on program correctness [5], especially chapters 5 and 7. The (ultra)metric distances defined between sets, and convergence with respect to such metrics we use, are also extensively employed by NIVAT and ARNOLD ([13] and [2]) considering among other subjects, infinite trees and nondeterminism. In their approach, trees are essentially programs, whereas we use trees of states, i.e. traces of program executions. Furthermore, Arnold and Nivat obtain the

set of all trees by completion of the set of all finite trees. We describe a tree by the set of all paths in the tree; the set of all trees is the set of all paths restricted in a suitable way (cf. Definition 9).

It appears that at three stages of the development we are forced to make the same restriction on the set of sequences used. This restriction amounts to require a tree-like property with respect to the occurrence of infinite branches.

This central tree-like property already was observed by R.-J. BACK in [3] treating unbounded nondeterminism.

The setup of the paper is as follows. After this introduction, in chapter 1 the syntax and some preliminary information are given. Chapter 2 starts with the definition of an operational semantics by means of the function *Comp*, which in turn is defined by a set of equations. The main result here is the existence proof of a unique solution *Comp* of this set of equations. In chapter 3 a denotational semantics is described concisely. Finally, in chapter 4 we prove the equivalence of the operational semantics to the denotational one.

1. SYNTAX AND PRELIMINARIES

Recursive procedures and finite nondeterministic choice are the key characteristics of the chosen language. Note, that subscripted variables are not treated (i.e., no arrays are present). Including these would necesitate a more complicated framework and only obscure our intentions. A straightforward extension is possible. The phrase "Let $(\alpha\epsilon)C$ be specified by $\alpha ::= qr|x|\alpha_1\alpha_2$ is to be understood as: All α or α_i, $i \in I$ in the sequel are assumed to be elements of the set C; α is of the form qr or x or $\alpha_1\alpha_2$, where α_1, α_2 are elements of C already.

We now define the sets of the syntactic entities we use.

Definition 1 Syntax

Let $(x\epsilon)$ *Ivar* be the set of integer variables

Let $(m\epsilon)$ *Icon* be the set of constants

Let $(P\epsilon)$ *Pvar* be the set of procedure variables

Let ($t\in$) *Iexp* be the set of integer expressions specified by

$t::=x|m|t_1+t_2|\ldots|$ if b then t_1 else t_2 fi

Let ($b\in$) *Bexp* be the set of boolean expressions specified by

$b::=true|false|t_1=t_2|\ldots|\neg b|b_1 \supset b_2$

Let ($S\in$) *Stat* be the set of statements, specified by

$S::=x:=t|S_1;S_2|S_1 \vee S_2|$ if b then S_1 else S_2 fi$|P$

Let ($E\in$) *Decl* be the set of declarations, specified by

$E::=\langle P_i \Leftarrow S_i \rangle_{i=1}^{n}$, $n\geq 0$, $P_i \neq P_j$, $1 \leq i < j \leq n$

Let ($R\in$) *Prog* be the set of programs, specified by

$R::=\langle E|S\rangle$, for all P in S or S_i, i=1,...,n, there exists j, $1 \leq j \leq n$ such that $P \equiv P_j$;

Note, that bounded choice now can be obtained by applying $(S_1 \vee S_2) \vee S_3$. The instances left open in *Iexp* and *Bexp* can be filled in with analogous expressions. Note that *Prog* is defined such, that all programs are closed, i.e. only these procedure variables occur in a program, for which the procedure body is given in the declaration E.

The following definitions concern assigning meaning to syntactic objects, i.e. semantics. At this stage, there is no distinction between operational and denotational semantics. Meaning is assigned by way of functions, defined by cases, from a syntactic domain to a domain of interpretation. To enable us later to define the rest of the denotational semantics we design the domains of interpretation as complete partial orders (cpo's).

DEFINITION 2. $(C,\sqsubseteq)$ is a cpo iff

(i) $\sqsubseteq$ is a partial order on C.

(ii) there is an element $\perp \in C$ such that, for all $c \in C$, $\perp \sqsubseteq c$

(iii) each chain $\langle c_i \rangle_{i=1}^{\infty}$ has a least upper bound $\bigsqcup_{i=1}^{\infty} c_i \in C$.

DEFINITION 3. Domains of interpretation.

$V_0 = \mathbb{N}$, natural numbers

$W_0 = \{tt,ff\}$, truth values

$\Sigma_0 = Ivar \to V_0$, functions assigning meaning to variables

Let $(\alpha\in)$ $V = V_0 \cup \{\perp_v\}$, cpo by $\alpha_1 \sqsubseteq \alpha_2$ iff $\alpha_1 = \perp_v$ or $\alpha_1 = \alpha_2$

Let $(\beta\in)$ $W = W_0 \cup \{\perp_w\}$, cpo analogously

Let $(\sigma\in)$ $\Sigma = \Sigma_0 \cup \{\perp\}$, cpo analogously

DEFINITION 4. For C cpo, $c_1, c_2, \perp_c \in C$

$$\text{if } \beta \text{ then } c_1 \text{ else } c_2 \text{ fi} = \begin{cases} c_1, & \text{if } \beta = tt \\ c_2, & \text{if } \beta = ff \\ \perp_c, & \text{if } \beta = \perp_w \end{cases}$$

We now define the meaning functions for integer and boolean expressions which yield, by cases, for each of the expressions and a state σ a value in one of the domains of interpretation.

DEFINITION 5.

(a) $\mathcal{V}$: $Iexp \to (\Sigma \to V)$

$\mathcal{V}(t)(\perp) = \perp_v$

For $\sigma \in \Sigma_0$

$\mathcal{V}(x)(\sigma) = \sigma(x)$

$\mathcal{V}(m)(\sigma) = \alpha$ where α is the integer denoted by m.

$\mathcal{V}(t_1+t_2)(\sigma) = \mathcal{V}(t_1)(\sigma) + \mathcal{V}(t_2)(\sigma)$

...

$\mathcal{V}$ (if b then t_1 else t_2 fi)(σ) = if $W(b)(\sigma)$ then $V(t_1)(\sigma)$ else $V(t_2)(\sigma)$ fi

(b) $\mathcal{W}$: $Bexp \to (\Sigma \to W)$

$\mathcal{W}(b)(\perp) = \perp_w$

For $\sigma \in \Sigma_0$

$\mathcal{W}(\text{true})(\sigma) = tt$

$\mathcal{W}(\text{false})(\sigma) = ff$

$\mathcal{W}(t_1=t_2)(\sigma) = (\mathcal{V}(t_1)(\sigma) = \mathcal{V}(t_2)(\sigma))$

...

$\mathcal{W}(\neg b)(\sigma) = \neg\mathcal{W}(b)(\sigma)$

$\mathcal{W}(b_1 \supset b_2)(\sigma) = (\mathcal{W}(b_1)(\sigma) \Rightarrow \mathcal{W}(b_2)(\sigma))$

We end this chapter by introducing the notion variant of a state. The purpose of this is to be able to indicate the effect of executing a statement, for instance an assignment statement x:=t by a change in the state. The following definition enables us to change in a state σ the value σ assigns to a particular x.

DEFINITION 6.

$$\perp\{\alpha/x\} = \perp$$

$$\sigma\{\alpha/x_1\}(x_2) = \begin{cases} \alpha, & \text{if } x_1 \equiv x_2 \\ \sigma(x_2), & \text{if } x_1 \not\equiv x_2 \end{cases}$$

2. THE OPERATIONAL SEMANTICS

The aim here is to define an operational semantics which is intuitively close to the actual program execution.

In the deterministic case a well-known way to achieve this is by way of a COOK semantics [7]. A function *Comp* yields for each program R and each state σ a, possibly infinite, computation sequence of states, $Comp(R)(\sigma) = \langle\sigma_1,\sigma_2,\ldots\rangle$. Intuitively, these states correspond to the states a computer goes through when executing R, starting in σ. The operational semantics then is a function *O* which yields for each program R and each state σ the state $\kappa(Comp(R)(\sigma))$, this being the last element of $Comp(R)(\sigma)$ if this sequence is finite and the special state $\perp$ otherwise.

Now intuitively *Comp* should be defined by rules, stepwise generating the computation sequences; a COOK semantics does so by cases, the cases being possible program forms. For example,

$$Comp(\langle E|S_1;S_2\rangle)(\sigma) = \langle\sigma\rangle\hat{}\,Comp(\langle E|S_1\rangle)(\sigma)\hat{}$$

$$Comp(\langle E|S_2\rangle)(\kappa(Comp(\langle E|S_1\rangle)(\sigma))).$$

The $\langle\sigma\rangle$ is motivated as to indicate the operation of splitting up $S_1;S_2$, or as a means to make induction arguments later on go through.

Adding nondeterminism necessitates *Comp* to yield for each R and σ not the corresponding computation sequence, but the set of computation sequences covering all possible alternatives depending on the different possible choices. We now define computation sequences and a set of rules to describe *Comp*.

DEFINITION 7. Computation sequences

(a) $\Sigma^+ = \{\langle\sigma_1,\ldots,\sigma_i,\ldots,\sigma_n\rangle \mid \sigma_i \in \Sigma,\ 1 \le i \le n,\ n \in \mathbb{N}\}$

$\Sigma^\omega = \{\langle\sigma_1,\ldots,\sigma_i,\ldots\rangle \mid \sigma_i \in \Sigma,\ i \in \mathbb{N}\}$

$\Sigma^\infty = \{\rho,\ldots\} = \Sigma^+ \cup \Sigma^\infty$

Note, that the empty sequence is excluded.

(b) $\hat{}: \Sigma^\infty \times \Sigma^\infty \to \Sigma^\infty$, concatenation, is defined by

$\langle\sigma_1,\ldots,\sigma_m\rangle\hat{}\langle\sigma_{m+1},\ldots,\sigma_n\rangle = \langle\sigma_1,\ldots,\sigma_m,\sigma_{m+1},\ldots,\sigma_n\rangle$

$\langle\sigma_1,\ldots,\sigma_m\rangle\hat{}\langle\sigma_{m+1},\ldots\rangle = \langle\sigma_1,\ldots,\sigma_m,\sigma_{m+1},\ldots\rangle$

$\langle\sigma_1,\ldots\rangle\hat{}\rho = \langle\sigma_1,\ldots\rangle$

with the extension $\rho\hat{}\{\rho_i \mid i \in I\} = \{\rho\hat{}\rho_i \mid i \in I\}$.

(c) $\kappa: \Sigma^\infty \to \Sigma$ is defined by

$$\kappa(\rho) = \begin{cases} \text{last element of } \rho, & \text{if } \rho \in \Sigma^+ \\ \perp, & \text{otherwise} \end{cases}$$

with the extension: $\kappa(\{\rho_i \mid i \in I\}) = \{\kappa(\rho_i) \mid i \in I\}$.

(d) $$\text{length}(R) = \begin{cases} n, & \text{if } \rho = \langle\sigma_1,\ldots,\sigma_n\rangle \\ \infty, & \text{otherwise.} \end{cases}$$

(e) ρ' is initial segment of ρ (i.s.o.) iff $\rho = \rho'\hat{}\rho''$ or $\rho = \rho'$.

In the sequel, $P(\Sigma^\infty) = \{A \mid A \subset \Sigma^\infty\}$, the powerset of Σ^∞.

DEFINITION 8. Rules for generating computation sequences.

Comp: *Prog* $\to (\Sigma \to P(\Sigma^\infty))$ by: For all $R \in$ *Prog*,

for $\sigma = \perp$, $Comp(R)(\perp) = \{\langle\perp\rangle\}$, for $\sigma \in \Sigma_0$

(i) $Comp(\langle E|x:=t\rangle)(\sigma) = \{\langle\sigma\{V(t)(\sigma)/x\}\rangle\}$

(ii) $Comp(\langle E|S_1;S_2\rangle)(\sigma) = \cup\{\langle\sigma\rangle\hat{}\rho\hat{}Comp(\langle E|S_2\rangle)(\kappa(\rho))|\rho \in Comp(\langle E|S_1\rangle)(\sigma)\}$

(iii) $Comp(\langle E|S_1 v S_2\rangle)(\sigma) = \langle\sigma\rangle\hat{}Comp(\langle E|S_1\rangle)(\sigma) \cup \langle\sigma\rangle\hat{}Comp(\langle E|S_2\rangle)(\sigma)$

(iv) $Comp(\langle E \mid$ if b then S_1 else S_2 fi$\rangle)(\sigma) =$

$= $ if $W(b)(\sigma)$ then $\langle\sigma\rangle\hat{}Comp(\langle E|S_1\rangle)(\sigma)$ else $\langle\sigma\rangle\hat{}Comp(\langle E|S_2\rangle)(\sigma)$ fi

(v) $Comp(\langle E|P\rangle)(\sigma) = \langle\sigma\rangle\hat{}Comp(\langle E|S_j\rangle)(\sigma)$, where $P \equiv P_j$, $P_j \Leftarrow S_j$ in E.

Intuitively, these rules are sufficient to describe generating the set of computation sequences for given R and σ. However the concept "generating" is too fuzzy to be mathematically satisfying. A well-known way out of this problem is to regard this set of rules as a set of equations, for which *Comp* should be a solution. From now on we take this approach: Definition 8 is regarded as a set of equations. Now it is clear that then a proof is required that a solution *Comp* exists, and moreover that it is unique. For the deterministic case this id done in various ways by DE BRUIN in [6]. For the nondeterministic case we now show that an extra equation is needed to ensure uniqueness. We then prove the existence of a unique solution *Comp* by extending the techniques of [6]. Then finally we define the operation semantics.

The following examples show, that in general, DEFINITION 8 regarded as a set of equations does not ensure a solution to be unique and provide intuition as to which kind of extra equation might solve this deficiency.

EXAMPLE 1. $Comp(\langle P \Leftarrow P|P\rangle)(\sigma)$. Intuitively, this should generate $\{\langle\sigma,\sigma,...\rangle\}$. However, regarded as an equation, this program gives rise to

$$Comp(\langle E|P\rangle)(\sigma) = \langle\sigma\rangle\hat{}Comp(\langle E|P\rangle)(\sigma).$$

Now both $\{\langle\sigma,\sigma,...\rangle\}$ and $\emptyset$ satisfy this equation, as Definition 7b implies $\rho\hat{}\emptyset = \emptyset$, so uniqueness is violated. Both practice (the program will loop) and theory (the rules generate $\langle\sigma,\sigma,...\rangle$) suggest preference for the first alternative.

The above example suggests the extra equation to be of the form $Comp(R)(\sigma) \neq \emptyset$. However, the following example shows that a stronger requirement is needed.

EXAMPLE 2.

$$Comp(\langle P \Leftarrow x:=x \vee P \mid P\rangle)(\sigma)$$

Intuitively, this should generate $\{\langle\sigma,\sigma,\sigma\rangle,\langle\sigma,\sigma,\sigma,\sigma,\sigma\rangle,\ldots,\langle\sigma,\sigma,\ldots\rangle\} = C$ respectively for x:=x chosen the first time possible, the second time, ..., never. However, regarded as an equation this program gives rise to

$$Comp(\langle E \mid P\rangle)(\sigma) = \langle\sigma\rangle\hat{}Comp(\langle E \mid x:=x \vee P\rangle)(\sigma)$$

$$= \langle\sigma\rangle\hat{}\langle\sigma\rangle\hat{}Comp(\langle E \mid x:=x\rangle)(\sigma) \cup \langle\sigma\rangle\hat{}\langle\sigma\rangle\hat{}Comp(\langle E \mid P\rangle)(\sigma)$$

$$= \{\langle\sigma,\sigma,\sigma\rangle\} \cup \langle\sigma,\sigma\rangle\hat{}Comp(\langle E \mid P\rangle)(\sigma)$$

Now both C and $C\setminus\{\langle\sigma,\sigma,\ldots\rangle\}$ satisfy this equation, so uniqueness of the solution is violated. Both practice and theory indicate which one should be preferred. Considering practice, a cycle that halts or is repeated according to nondeterministic choice potentially can be repeated any finite amount of time, and also can be repeated forever. So this suggests preference for the first alternative. Considering theory, for the obvious representation of the set of computation sequences by trees, finite nondeterministic choice gives rise to finitely branching trees. By König's lemma then follows that a tree containing infinitely many finite branches, i.e. fintie computation sequences, also contains an infinite one, so this also suggests preference for the first alternative.

The above examples suggest the entire equation to be of the form: $Comp(R)(\sigma) \in G$, where $G = \{G \in \mathcal{P}(\Sigma^\infty) \mid G \neq \emptyset$, if $\langle\rho_i\rangle_{i=1}^\infty$ such that

(i) for all i, $\rho_i \in G$

(ii) for all i, ρ_i i.s.o. ρ_{i+1}

(iii) $\sup_i\{length(\rho_i)\} = \infty$

then $\exists\rho \in G$ such that for all i, ρ_i, i.s.o. $\rho\}$

However, the following example shows that an even more subtle requirement is needed.

EXAMPLE 3.

$$Comp(\langle P \Leftarrow x:=1 \vee P \mid P\rangle)(\sigma)$$

Intuitively, this should generate $\{\langle\sigma,\sigma,\sigma\{1/x\}\rangle,\langle\sigma,\sigma,\sigma,\sigma,\sigma\{1/x\}\rangle,\ldots,\langle\sigma,\sigma,\ldots$ $\ldots\rangle\} = C'$. However, regarded as an equation, like in example 2, $C'\setminus\{\langle\sigma,\sigma,\ldots\rangle\}$ is also possible. Again, the first alternative is to be preferred.

This example suggests the following strengthening of the above chosen requirement described by G.

DEFINITION 9.
$H = \{H \in P(\Sigma^\infty) \mid H \neq \emptyset$, if $\langle\rho_i\rangle_{i=1}^\infty$, $\rho_i \in \Sigma^\infty$, such that
(i) for all i, $\exists\rho_i' \in H$ such that ρ_i i.s.o. ρ_i'
(ii) for all i, ρ_i i.s.o. ρ_{i+1}
(iii) $\sup\{\text{length}(\rho_i)\} = \infty$
then $\exists\rho \in H$ such that for all i, ρ_i i.s.o. $\rho\}$

REMARK. In the different setting of unbounded nondeterminism, this the closedness property to be found in [3].

We claim that the following extension of Definition 8 ensures the existence of a unique solution.

DEFINITION 10. $Comp$: $Prog \to (\Sigma \to H)$ is defined by the following set of equations:
(a) The equations of Definition 8.
(b) For all $R \in Prog$, $\sigma \in \Sigma$, $Comp(R)(\sigma) \in H$.

In DE BRUIN [6] for the deterministic case four methods to prove the existence of a unique solution are presented. We have chosen to adapt to our case the most topological one, as this seems the best one to extend to sets of sequences. The idea is the following. Consider the set of functions $Prog \to (\Sigma \to H)$; the solution $Comp$ we seek to find is, if it exists, an element

of this set. Now as the left parts of the equations in part (a) of Definition 10 all contain only $Comp(R)(\sigma)$, a solution of this set of equations can be interpreted as a fixed point of an endomorphic operator on $Prog \to (\Sigma \to H)$ defined directly by these equations (cf. Definition 20). To ensure existence of a unique fixed point, from topology it is known that it is sufficient that firstly the space is complete metric, i.e. a space with a metric distance function defined on it such that every Cauchy sequence converges, and secondly, that the operator is a contraction mapping, i.e. the distance between the image of any two points is less than or equal to the distance between the original points multiplied by a fixed constant smaller than 1.

The operator as well as the elements of the domain are given: respectively by the equations and by the type of *Comp*. Left to choose is the metric. As usual, we choose the distance between two functions to be the supremum over the elements in the domain of the distance between the two images of such an element. To do so, we first define a distance between computation sequences, next between sets of them and finally between the functions. All of these will have to be complete metrics.

We start by considering computation sequences, i.e. Σ^∞. A natural distance is the following.

DEFINITION 11.

$$\rho[j] = \begin{cases} \rho, & \text{if } \rho = \langle\sigma_1,\ldots,\sigma_n\rangle \text{ and } j \geq n \\ \langle\sigma_1,\ldots,\sigma_j\rangle, & \text{otherwise} \end{cases}$$

DEFINITION 12. Distance on Σ^∞

$$\bar{d}(\rho_1,\rho_2) = \begin{cases} 2^{-k}, & \text{if } k = \sup\{j \mid \rho_1[j] = \rho_2[j]\} < \infty \\ 0, & \text{otherwise} \end{cases}$$

DEFINITION 13. For a Cauchy sequence $\langle\rho_i\rangle_{i=1}^\infty$ define the limit $\lim_{i\to\infty} \rho_i$ as follows. As $\langle\rho_i\rangle_{i=1}^\infty$ has the Cauchy property, $\forall\varepsilon > 0 \exists N_\varepsilon \forall \ell,m \geq N_\varepsilon d(\rho_\ell,\rho_m) < \varepsilon$, or, equivalently, $\forall k \in \mathbb{N} \exists N_k \forall \ell,m \geq N_k \bar{d}(\rho_\ell,\rho_m) < 2^{-k}$.

By Definition 12 this implies $\forall k \in \mathbb{N}\, \exists N_k \forall \ell, m \geq N_k \rho_{N_k}[k] = \rho_\ell[k] = \rho_m[k]$. Now define $\lim_{i\to\infty} \rho_i$ by $(\lim_{i\to\infty} \rho_i)[k] = \rho_{N_k}[k]$.

LEMMA 1. $(\Sigma^\infty, \bar{\bar{d}})$ *is a complete metric space.*

PROOF. $\bar{\bar{d}}$ evidently is a metric. $\bar{\bar{d}}$ is complete iff every Cauchy sequence converges. Clearly, every Cauchy sequence $\langle \rho_i \rangle_{i=1}^\infty$ converges to $\lim_{i\to\infty} \rho_i$. □

Next, we proceed to sets of computation sequences. Note, that defining the distance $\bar{\bar{d}}$ enables us to give a much easier definition of H.

LEMMA 2. $H = \{H \in P(\Sigma^\infty) \mid H \neq \emptyset$, *for each Cauchy sequence* $\langle \rho_i \rangle_{i=1}^\infty$ *in* H, $\lim_{i\to\infty} \rho_i \in H\}$.

PROOF. Evident by Definitions 9, 12 and 13. □

REMARK.

(1) Here the topological approach allows an easier solution of the problem than the cpo approach, where it is more difficult to handle cases like example 3 as may be seen by the difference between the two definitions of H.

(2) For Σ^∞ with the topology $J(\bar{\bar{d}})$ induced by $\bar{\bar{d}}$, H can be defined by $H = \{H \in P(\Sigma^\infty) \mid H \neq \emptyset$, H closed in $J(\bar{\bar{d}})\}$.

A natural distance on H is defined as follows.

DEFINITION 14.

$$H[j] = \{\rho[j] \mid \rho \in H\}.$$

DEFINITION 15. Distance on H.

$$\bar{d}(H_1, H_2) = \begin{cases} 2^{-k}, & \text{if } k = \sup\{j \mid H_1[j] = H_2[j]\} < \infty \\ 0, & \text{otherwise.} \end{cases}$$

DEFINITION 16. For a Cauchy sequence $\langle H_i \rangle_{i=1}^{\infty}$ define the limit $\lim_{i\to\infty} H_i$ as follows. As $\langle H_i \rangle_{i=1}^{\infty}$ has the Cauchy property, $\forall \varepsilon > 0 \exists N_\varepsilon \forall \ell,m \geq N_\varepsilon \bar{d}(H_\ell,H_m) < \varepsilon$, or, equivalently, $\forall k \in \mathbb{N} \exists N_k \forall \ell,m \geq N_k \bar{d}(H_\ell,H_m) < 2^{-k}$. By Definition 15 this implies $\forall k \in \mathbb{N} \exists N_k \forall \ell,m \geq N_k H_{N_k}[k] = H_k[k] = H_m[k]$. Now define $\lim_{i\to\infty} H_i$ as follows (using Lemma 2).

$$\lim_{i\to\infty} H_i = \bigcup_{k=1}^{\infty} \{\rho \in H_{N_k}[k] \mid \rho = \langle \sigma_1,\ldots,\sigma_n \rangle,\ n < k\}$$

$$\cup \{\lim_{k\to\infty} \rho_k \mid \langle p_k \rangle_{k=1}^{\infty} \text{ Cauchy sequence, } \rho_k \in H_{N_k}[k]\}$$

LEMMA 3. *$(H,\bar{d})$ is a complete metric space.*

PROOF. The first requirement to be a metric space is $\bar{d}(H_1,H_2) = 0 \iff H_1 = H_2$ Let $\bar{d}(H_1,H_2) = 0$, $\rho \in H_1$. If $\rho \in \Sigma^+$, then $\exists j\ \rho = \rho[j] = \rho[j+1]$. As $\bar{d}(H_1,H_2) = 0$, $\rho = \rho[j] = \rho[j+1] \in H_2[j+1]$. Consequently, $\rho \in H_2$. If $\rho \in \Sigma^\omega$, then either $\rho \in H_2$ or $\rho[i] \in H_2[i]$, $i = 1,2,\ldots$. In the latter case, there exist $\rho_i' \in H_2$, $i = 1,2,\ldots$, such that $\rho[i] = \rho'[i]$, $i = 1,2,\ldots$. Now clearly $(\rho_i')_{i=1}^{\infty}$ is a Cauchy sequence in H_2, and by Definition 13 $\lim_{i\to\infty} \rho_i' = \rho$. Consequently, (by Lemma) $\rho \in H_2$. Conversely let $H_1 = H_2$. Then $\forall_j\ H_1[j] = H_2[j]$, so $d(H_1,H_2) = 0$. The other requirements of being a metric space are evidently fulfilled. $\bar{d}$ is complete iff every Cauchy sequence converges. Let $\langle H_i \rangle_{i=1}^{\infty}$ be a Cauchy sequence. By Definition 16, $\langle H_i \rangle_{i=1}^{\infty}$ clearly converges to $\lim_{i\to\infty} H_i$, by Definition 16, clearly $\lim_{i\to\infty} H_i \in H$. □

REMARK. The reasons to restrict $P(\Sigma^\infty)$ to H in the beginning of this chapter that did arise when regarding Definition 8 as a set of equations here have their topological counterpart: Should distance $\bar{d}$ be defined on $P(\Sigma^\infty)$ instead of H, then the sets C and C\{<σ,σ,...>} of Example 2 (and likewise C' and C'\{<σ,σ,...>} of Example 3) would have distance 0 but not be equal.

This violates the metric requirement $\bar{d}(C_1,C_2) = 0 \iff C_1 = C_2$. Now disregard knowledge of the previously defined restriction of $P(\Sigma^\infty)$ to H caused by ambiguities with regard to solutions of the equations in Definition 1 and indicated by the Examples 1 - 3. (Note, that at that stage no distance between sets was even defined.) A natural solution of the present problem then, is the following.

Restrict $P(\Sigma^\infty)$ to only those subsets of Σ^∞, that contain their limit points in the topology induced by $\bar{d}$. Lemma 3 states that this solves the problem. Not surprisingly, the tree-likeness requirement stated in H is equivalent to this restriction, as stated in Lemma 2.

By now, we arrive at our first aim, turning $Prog \to (\Sigma \to H)$ into a complete metric space by defining the following natural distance.

DEFINITION 17.

$$(\phi\epsilon)C = Prog \to (\Sigma \to H).$$

DEFINITION 18. Distance on C.

$$d(\phi_1,\phi_2) = \sup_{R,\sigma}\{\bar{d}(\phi_1(R)(\sigma),\ \phi_2(R)(\sigma))\}.$$

DEFINITION 19. For a Cauchy sequence $<\phi_i>_{i=1}^\infty$ define the limit $\lim_{i\to\infty} \phi_i$ as follows. As $<\phi_i>_{i=1}^\infty$ has the Cauchy property, $\forall\varepsilon > 0\ \exists N_\varepsilon \forall \ell,m \geq N_\varepsilon d(\phi_\ell,\phi_m) < \varepsilon$. By Definition 18 holds $\forall\varepsilon > 0\ \exists N_\varepsilon \forall \ell,m \geq N_\varepsilon \forall R \forall\sigma\ d(\phi_\ell(R)(\sigma),\ \phi_m(R)(\sigma)) < \varepsilon$. Then $\forall R\forall\sigma$, $<\phi_i(R)(\sigma)>_{i=1}^\infty$ is a Cauchy sequence. By Lemma 3, $\forall R\forall\sigma<\phi_i(R)(\sigma)>_{i=1}^\infty$ converges to $\lim_{i\to\infty} \phi_i(R)(\sigma)$. Now define $\lim_{i\to\infty} \phi_i$ as follows.

$$\forall R\forall\sigma(\lim_{i\to\infty} \phi_i)(R)(\sigma) = \lim_{i\to\infty}(\phi_i(R)(\sigma)).$$

LEMMA 4. (C,d) *is a complete metric space.*

PROOF. By standard techniques, e.g. see ([8], Chapter 14 Theorem 2.6).

DEFINITION 20.

Φ: C $\to$ C is defined by

$\Phi = \lambda\phi\cdot\lambda R\cdot\lambda\sigma\cdot\sigma = \bot \to \bot$

$\sigma \in \Sigma_0$,

$R \equiv \langle E|x:=t\rangle \to \{\langle\sigma\{V(t)(\sigma)/x\}\rangle\}$

$R \equiv \langle E|S_1;S_2\rangle \to$
$\to \langle\sigma\rangle\hat{}\phi(\langle E|S_1\rangle)(\sigma)\hat{}\phi(\langle E|S_2\rangle)(\kappa(\phi(\langle E|S_1\rangle)(\sigma)))$

$R \equiv \langle E|S_1 \vee S_2\rangle \to \langle\sigma\rangle\hat{}\phi(\langle E|S_1\rangle)(\sigma) \cup \langle\sigma\rangle\hat{}\phi(\langle E|S_2\rangle)(\sigma)$

$R \equiv \langle E \mid$ if b then S_1 else S_2 fi$\rangle \to$
if $W(b)(\sigma)$ then $\langle\sigma\rangle\hat{}\phi(\langle E|S_1\rangle)(\sigma)$
else $\langle\sigma\rangle\hat{}\phi(\langle E|S_2\rangle)(\sigma)$

$R \equiv \langle E|P\rangle \to \langle\sigma\rangle\hat{}\phi(\langle E|S_j\rangle)$, where $P \equiv P_j$, $P_j \Leftarrow S_j$ in E.

Note, that Φ is well-defined, i.e. $\forall\phi\ \Phi(\phi) \in C$, as can be easily seen from the definition.

LEMMA 5. *Φ is a contraction mapping, namely* $\forall\phi_1,\phi_2\ d(\Phi(\phi_1),\Phi(\phi_2)) \leq \frac{1}{2}d(\phi_1,\phi_2)$.

PROOF. Each of the following cases is trivial for $\sigma = \bot$, so from here on $\sigma \in \Sigma_0$.

Case 1. $R \equiv \langle E|x:=t\rangle$

By Definition 20

$$\forall\phi_1,\phi_2\ \forall\sigma\Phi(\phi_1)(R)(\sigma) = \langle\sigma\{V(t)(\sigma)/x\}\rangle\} = \Phi(\phi_2)(R)(\sigma).$$

So by Definitions 15 and 18, $\bar{d}(\Phi(\phi_1),\Phi(\phi_2)) = 0 \leq \frac{1}{2}\bar{d}(\phi_1,\phi_2)$.

Case 2. $R \not\equiv \langle E|x:=t\rangle$, $R \not\equiv \langle E|S_1;S_2\rangle$.

By Definition 20, $\forall\phi\forall\sigma\ \Phi(\phi)(R)(\sigma) = \langle\sigma\rangle\hat{}\phi(R')(\sigma)$, R' as given by the right hand part of definition 20.

So

$$d(\Phi(\phi_1),\Phi(\phi_2)) = \sup_{R,\sigma}\{\bar{d}(\Phi(\phi_1)(R)(\sigma),\Phi(\phi_2)(R)(\sigma))\}, \text{ by definition 18}$$

$$= \sup_{R,\sigma}\{\bar{d}(\langle\sigma\rangle\hat{}\phi_1(R')(\sigma),\langle\sigma\rangle\hat{}\phi_2(R')(\sigma))\}$$

$$= \tfrac{1}{2}\sup_{R,\sigma}\{\bar{d}(\phi_1(R')(\sigma),\phi_2(R')(\sigma))\}, \text{ by definition 15}$$

$$= \tfrac{1}{2}\, d(\phi_1,\phi_2), \text{ by definition 18.}$$

Case 3. $R \equiv \langle E|S_1;S_2\rangle$, analogously to case 2. □

By now we can, by using well-known topology, justify our claim made above.

LEMMA 6. *Φ has a unique fixpoint.*

PROOF. By Lemmas 4 and 5, using standard techniques from topology. Cf. ([8], p. 305) and ([6], p.2). □

THEOREM 1. *The set of equations of Definition* 10 *has a unique solution* Comp.

PROOF. Directly from Definitions 10 and 20 and Lemma 6. □

Finally, having justified the definition of *Comp*, we define the operational semantics.

DEFINITION 21. Operational semantics. $\mathcal{O}$: *Prog* $\to (\Sigma \to \Sigma)$ is defined by: For all R, for all σ, $\mathcal{O}(R)(\sigma) = \kappa(Comp(R)(\sigma))$.

For later use we here state the following lemma.

LEMMA 7.

(i) $\mathcal{O}(\langle E|S_1;S_2\rangle)(\sigma) = \mathcal{O}(\langle E|S_2\rangle)\circ\mathcal{O}(\langle E|S_1\rangle)(\sigma)$

(ii) $\mathcal{O}(\langle E|S_1 \vee S_2\rangle)(\sigma) = \mathcal{O}(\langle E|S_1\rangle)(\sigma) \cup \mathcal{O}(\langle E|S_2\rangle)(\sigma)$

(iii) $\mathcal{O}(\langle E \mid$ if b then S_1 else S_2 fi$\rangle)(\sigma) =$ if $\mathcal{W}(b)(\sigma)$ then $\mathcal{O}(\langle E|S_1\rangle)(\sigma)$ else $\mathcal{O}(\langle E|S_2\rangle)(\sigma)$ fi

(iv) $\mathcal{O}(\langle E|P\rangle)(\sigma) = \mathcal{O}(\langle E|S_j\rangle)(\sigma)$, where $P \equiv P_j$, $P_j \Leftarrow S_j$ in E.

PROOF. Evident from Definitions 10 and 21. □

3. THE DENOTATIONAL SEMANTICS

We here present the denotational semantics for which the operational one, treated in Chapter 2, was designed to serve as an intuitive counterpart. The method used is the fixed point approach in a cpo setting, as can be found in [15]. The denotational semantics we use greatly resembles the one in ([5], Chapters 5 and 7), so only a very concise treatment is given, just

defining the notions and stating the results we need for the equivalence proof in Chapter 4.

We start by defining a domain, consisting of a selection of subsets of $P(\Sigma)$ with the Egli-Milner ordering, cf. [10]. Note the resemblance to the domain of results in Chapter 2, with regard to H being the outcome domain of *Comp*.

DEFINITION 22.

$$(\tau\in)T = \{\tau \in P(\Sigma) \mid \tau \text{ finite or } \perp \in \tau\}$$

DEFINITION 23. Egli-Milner ordering

$$\tau_1 \sqsubseteq \tau_2 \text{ iff } \perp \in \tau_1 \text{ and } \tau_1\backslash\{\perp\} \subseteq \tau_2$$

$$\text{or } \perp \notin \tau_1 \text{ and } \tau_1 = \tau_2.$$

LEMMA 8. $(T,\sqsubseteq)$ *is a cpo.*

We now give the domain of strict functions $\Sigma \to_s T$.

DEFINITION 24.

$$\psi: \Sigma \to_s T, \text{ i.e. } \psi \text{ is strict iff } \psi(\perp) = \{\perp\}.$$

DEFINITION 25.

$$(\psi\in)M = \Sigma \to_s T$$

with the extension: For each $\psi: \Sigma \to_s T$, $\hat{\psi}: T \to_s T$ by $\hat{\psi} = \lambda\tau\cdot \bigcup_{\sigma\in\tau} \psi(\sigma)$ and

$$\psi_1 \circ \psi_2 = \lambda\sigma\cdot\hat{\psi}_1(\psi_2(\sigma))$$

$$\psi_1 \cup \psi_2 = \lambda\sigma\cdot(\psi_1(\sigma) \cup \psi_2(\sigma)).$$

DEFINITION 26.

$$\psi_1 \sqsubseteq \psi_2 \text{ iff for all } \sigma \in \Sigma,\ \psi_1(\sigma) \sqsubseteq \psi_2(\sigma)$$

LEMMA 9 . $(M,\sqsubseteq)$ *is a cpo.*

We now introduce $\gamma \in \Gamma$, where γ gives meaning to procedure variables; furthermore we define variant of γ.

DEFINITION 27.

$$(\gamma\epsilon)\Gamma = Pvar \to M$$

DEFINITION 28.

$$\gamma\{\psi/P\}(P') = \begin{cases} \psi, & \text{if } P' \equiv P \\ \gamma(P'), & \text{otherwise} \end{cases}$$

The following is needed from the theory of cpo's.

DEFINITION 29. $(C,\sqsubseteq)$ cpo, $f: C \to C$

(a) x is a fixed point of f iff $f(x) = x$.

(b) x is the least fixed point μf of f iff:

(i) x is a fixed point of f;

(ii) for all y, y fixed point of f, $x \sqsubseteq y$.

DEFINITION 30. $(C,\sqsubseteq)$, $(C',\sqsubseteq)$ cpo; $f: C \to C'$ is continuous iff:

(i) $x_1 \sqsubseteq x_2 \Rightarrow f(x_1) \sqsubseteq' f(x_2)$ (monotonicity);

(ii) for each chain $\langle x_i \rangle_{i=0}^{\infty}$ in C,

$$f(\bigsqcup_{i=0}^{\infty} x_i) = \bigsqcup_{i=0}^{\infty} f(x_i)$$

Notation: $f \in [C \to C']$.

LEMMA 10. C_i cpo, $f_i \in [C^n \to C]$, $i = 1,\ldots,n$

$$\langle f_1,\ldots,f_n \rangle \in [C^n \to C^n]$$

by

$$\langle f_1,\ldots,f_n\rangle(\langle c_1,\ldots,c_n\rangle) = \langle f_1(\langle c_1,\ldots,c_n\rangle),\ldots,f_n(\langle c_1,\ldots,c_n\rangle)\rangle$$

Then

$$\mu\langle f_1,\ldots,f_n\rangle = \bigsqcup_{k=1}^{\infty} \langle f_1,\ldots,f_n\rangle^k(\langle \bot,\ldots,\bot\rangle)$$

After these preparations we define the denotational semantics as follows.

DEFINITION 31. Denotational semantics

(a) N: *Stat* $\to (\Gamma \to M)$ is defined by

(i) $N(x:=t)(\gamma) = \lambda\sigma\cdot\{\sigma\{V(t)(\sigma)/x\}\}$

(ii) $N(S_1;S_2)(\gamma) = N(S_2)(\gamma)\circ N(S_1)(\gamma)$

(iii) $N(S_1 \vee S_2)(\gamma) = N(S_1)(\gamma) \cup N(S_2)(\gamma)$

(iv) N (if b then S_1 else S_2 fi)$(\gamma) = \lambda\sigma\cdot$ if $W(b)(\sigma)$ then $N(S_1)(\gamma)(\sigma)$ else $N(S_2)(\gamma)(\sigma)$ fi

(b) M: *Prog* $\to (\Gamma \to M)$ is defined by

$M(\langle E|S\rangle)(\gamma) = N(S)(\gamma\{\psi_i/P_i\}_{i=1}^n)$, where $\langle\psi_1,\ldots,\psi_n\rangle = \mu\langle\Psi_1,\ldots,\Psi_n\rangle$ and $\Psi_j = \lambda\psi_1',\ldots,\lambda\psi_n'N(S_j)(\gamma\{\psi_i'/P_i\}_{i=1}^n)$, $j = 1,\ldots,n$.

LEMMA 11.

$$\lambda\psi_1',\ldots,\lambda\psi_n'\cdot N(S_j)(\gamma\{\psi_i'/P_i\}_{i=1}^n) \in [M^n \to M], \qquad j = 1,\ldots,n.$$

THEOREM 2. *M is well-defined.*

PROOF. Essentially from Lemmas 10 and 11. □

For later use, in Chapter 4, we here state the following lemmas.

LEMMA 12. $\lambda\sigma\cdot N(S)(\gamma)(\sigma)$ *is monotone.*

LEMMA 13. $M(\langle E|P\rangle) = M(\langle E|S_j\rangle)$, *where* $P \equiv P_j$, $P_j \Leftarrow S_j$ *in* E.

4. THE EQUIVALENCE OF THE OPERATIONAL AND THE DENOTATIONAL SEMANTICS

The set-up of the equivalence proof for the two semantics defined in the foregoing chapters, i.e. $O(R) = M(R)(\gamma)$, is as follows.

A natural way to proceed might seem to apply induction on the length of individual computation sequences in $Comp(R)(\sigma)$ proving

$\sigma' \in \mathcal{O}(R)(\sigma) \Rightarrow \sigma' \in M(R)(\gamma)(\sigma)$. However, it is only possible to prove this for σ' such that $Comp(R)(\sigma) \in P(\Sigma^+)$. Namely, if there is an infinite computation sequence in $Comp(R)(\sigma)$ then $\perp \in \mathcal{O}(R)(\sigma)$, as can be directly inferred from Definitions 7 and 21. It is by no means clear, that in this case also $\perp \in M(R)(\gamma)(\sigma)$, as the concept of computation sequence belongs to the realm of operational semantics. So using set inclusion $\mathcal{O}(R)(\sigma) \subseteq M(R)(\gamma)(\sigma)$ is not possible for this kind of inclusion. Choosing the Egli-Milner ordering $\mathcal{O}(R)(\sigma) \sqsubseteq M(R)(\gamma)(\sigma)$ with this induction is also impossible, as for this ordering it is required to prove $\mathcal{O}(R)(\sigma) = M(R)(\gamma)(\sigma)$ if $\perp \notin \mathcal{O}(R)(\sigma)$.

The way out we have chosen is to apply, in case $Comp(R)(\sigma) \in P(\Sigma^+)$, induction on the sum of the lengths of the computation sequences in $Comp(R)(\sigma)$, thus proving $\mathcal{O}(R)(\sigma) = M(R)(\gamma)(\sigma)$ in this case. In case there is an infinite computation sequence in $Comp(R)(\sigma)$, and so , by Definition 21, $\perp \in \mathcal{O}(R)(\sigma)$, we prove $\mathcal{O}(R)(\sigma)\setminus\{\perp\} \subseteq M(R)(\gamma)(\sigma)$ elementwise by the above mentioned induction on the length of individual computation sequences. Thus we yield $\mathcal{O}(R) \sqsubseteq M(R)(\gamma)$. Proving $M(R)(\gamma) \sqsubseteq \mathcal{O}(R)$ by standard techniques then completes the proof.

In order to apply induction to the sum of the lengths of the computation sequences in case $Comp(R)(\sigma) \in P(\Sigma^+)$ we have to prove that this sum is finite. This is made explicit by a careful application of an analogue of Königs lemma. One of the well-known formulations of König's lemma is the following:

LEMMA. (König's) *A finitely branching tree where all branches are of finite length contains only finitely many nodes.*

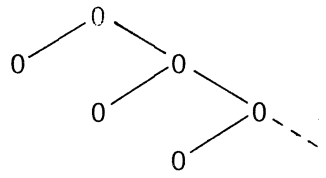
As we work in the realm of sets of (computation) sequences instead of trees, we want to restate this lemma using these notions. Restate "finitely branching" by "there are only finitely many different initial segments of length n, for all $n \in \mathbb{N}$", "all branches are of finite length" by "a set of finite sequences", and finally "finitely many nodes" by "finitely many sequences".

So the analogue to König's lemma seems to be

> If in a set of finite sequences there are only finitely many different initial segments of length n, for all $n \in \mathbb{N}$, then there are only finitely many different sequences.

Now this is not true!

Counter example: {<0>,<0,0>,...}. The reason for this is, that the tree structure does not allow {<0>,<0,0>,...} as a set of branches in a finitely branching tree but forces to add <0,0,...>:



For a set of finite sequences this is not the case. So an extra requirement of such a set is to be added. Not surprisingly, taking the set to have a property analogeous to *H* for computation sequences is sufficient, as this reflected the tree-like way in which *Comp* generated a 'set of computation sequences'.

We now give the analogue of König's lemma.

LEMMA 14. *If in a set* C *of finite sequences* {r,...}, r = $\langle s_1, s_2, \ldots, s_n \rangle$, n $\in \mathbb{N}$, *there are only finitely many different initial segments of length* n *for all* n $\in \mathbb{N}$, *and* C *has the following property:* C *is tree-like i.e. if there is a row of sequences* $\langle r_i' \rangle_{i=1}^{\infty}$, *not necessarily* $r_i' \in C$, *such that*

(i) *for all* i, $\exists r_i \in C$: r_i' i.s.o. r_i

(ii) *for all* i, r_i' i.s.o. r_{i+1}'

(iii) $\sup_i \{\text{length}(r_i')\} = \infty$

then $\lim_{i\to\infty} r_i' \in C$, *then there are only finitely many sequences in* C.

PROOF. By contradiction. Suppose there are infinitely many different sequences. Let G(n) = {r | length(r) = n, r i.s.o. infinitely many different sequences}. We show by induction that G(n) $\neq \emptyset$ for all n $\in \mathbb{N}$. Induction basis: To prove G(1) $\neq \emptyset$. As there are only finitely many different initial segments of length 1 and infinitely many different sequences, G(1) $\neq \emptyset$. Induction step: To prove G(k+1) $\neq \emptyset$. As there are infinitely many different sequences but only finitely many different initial segments of length k or k+1, and G(k) = $\emptyset$ (Ind. hyp.), G(k+1) $\neq \emptyset$. So G(n) $\neq \emptyset$ for all n $\in \mathbb{N}$. Now clearly, for all n $\in \mathbb{N}$ every element of G(n) is initial segment of at least one of the elements of G(n+1). So by the axiom of choice there are $\bar{r}_i \in G(i)$ such that $\bar{r}_i$ i.s.o. $\bar{r}_{i+1}$, i = 1,2,... . As C is tree-like, this implies that there is an infinite sequence in C. Contradiction. □

REMARK. Note that the property "tree-like" had to be brought to the surface on three fully independent occasions where it was more or less hidden in the structure of the concepts under consideration:

(1) In Definition 10 to select tree-like solutions of the equations.
(2) In Definition 15, restricting the distance $\bar{d}$ to a space consisting of only tree-like sets.
(3) In Lemma 14 to select sets sufficiently tree-like to prove an analogue of König's lemma for them.

We now show, that for all R and all σ, $Comp(R)(\sigma)$ satisfies the requirements of Lemma 14. The only requirement left to prove is, that $Comp(R)(\sigma)$ gives only rise to finitely many different initial segments. This is done in the following lemma.

LEMMA 15. *For all* R *and all* σ *the following holds for* $Comp(R)(\sigma)$*: There are only finitely many different initial segments of length* n*, for all* $n \in \mathbb{N}$*, in* $Comp(R)(\sigma)$.

PROOF. Let $R \equiv \langle E|S\rangle$. Proof by cases, applying induction on the length of the initial segment. Let

$$I(n)(Comp(R)(\sigma)) = \{\rho' \mid \rho' \text{ i.s.o. } \rho \in Comp(R)(\sigma), \text{ length } (\rho') = n\}$$

Induction basis: To prove $\#(I(1)(Comp(R)(\sigma))) < \infty$.

By cases:

(i) $S \equiv x:=t$. Then $Comp(R)(\sigma) = \{\langle\sigma\{V(t)(\sigma)/x\}\rangle\}$. Consequently, $\#(I(1)(Comp(R)(\sigma))) = 1 < \infty$.

(ii) $S \equiv S_1;S_2$. Then $Comp(R)(\sigma) = \cup\{\langle\sigma\rangle\hat{}\rho\hat{}Comp(\langle E|S_2\rangle)(\kappa(\rho)) \mid \rho \in Comp(\langle E|S_1\rangle)(\sigma)\}$, so $I(1)(Comp(R)(\sigma)) = \{\langle\sigma\rangle\}$. Consequently, $\#(I(1)(Comp(R)(\sigma))) = 1 < \infty$.

Cases (iii), (iv) and (v) of Definition 10 analogeously to (ii) lead to $\#(I(1)(Comp(R)(\sigma)) = 1 < \infty$. Induction hypothesis: Assume $\#(I(\ell)(Comp(R)(\sigma))) < \infty$, for $1 \leq \ell \leq k$. Induction step: To prove $\#(I(k+1)(Comp(R)(\sigma))) < \infty$.

By cases:

(i) $S \equiv x:=t$. Then $I(k+1)(Comp(R)(\sigma)) = I(k+1)(\{\langle\sigma\{V(t)(\sigma)/x\}\rangle\}) = \emptyset$.

Consequently, $\#(I(k+1)(Comp(R)(\sigma))) = 0 < \infty$.

(ii) $S \equiv S_1;S_2$. Then $Comp(R)(\sigma) = \{<\sigma>\hat{}\rho\hat{}Comp(<E|S_2>)(\kappa(\rho)) \mid \rho \in Comp(<E|S_1>)(\sigma)\}$. Consequently,

$$\#(I(k+1)(Comp(R)(\sigma))) = \#(I(k)(Comp(<E|S_1>)(\sigma))) +$$

$$+ \Sigma\{\#(I(k+1-(1+\text{length}(\rho)))(Comp(<E|S_2>)(\kappa(\rho)))) \mid$$

$$\rho \in Comp(<E|S_1>)(\sigma),\ \text{length}(\rho) < k\} < \infty \qquad \text{(Ind. hyp.)}$$

Cases (iii), (iv) and (v) of Definition 10 analogously to (ii) lead to $\#(I(k+1)(Comp(R)(\sigma))) < \infty$. So $\#(I(n)(Comp(R)(\sigma))) < \infty$ for all R, all σ, all $n \in \mathbb{N}$. □

After these preparations, we can state Lemma 16, which enables us to apply induction on the sum of the lengths of the computation sequences in $Comp(R)(\sigma)$ in case $Comp(R)(\sigma) \in P(\Sigma^+)$.

LEMMA 16. *For all* R *and all* σ *for which* $Comp(R)(\sigma) \in P(\Sigma^+)$, $Comp(R)(\sigma)$ *is a finite set.*

PROOF. It is given that all computation sequences in $Comp(R)(\sigma)$ are finite. By Lemma 15 there are only finitely many different initial segments of length n, for all $n \in \mathbb{N}$. By Definition 10b, and Definition 9, $Comp(R)(\sigma)$ has the tree-like property as required in Lemma 14. Consequently, by Lemma 14, $Comp(R)(\sigma)$ in this case is finite. □

Finally, we arrive at the main theorem of this chapter, stating equivalence of $O(R)$ and $M(R)(\gamma)$.

DEFINITION 32. For

$$A \in P(\Sigma^\infty), \text{length}(A) = \begin{cases} \Sigma\{\text{length}(\rho) \mid \rho \in A\}, & \text{if } \#(A) < \infty \text{ and,} \\ & \forall_\rho \in A,\ \rho \in \Sigma^+ \\ \infty & \text{otherwise.} \end{cases}$$

THEOREM 3. *For all* R *and all* γ, $O(R) = M(R)(\gamma)$.

PROOF. Let $R \equiv \langle E|S\rangle$. We prove $\forall R\ \forall\gamma\ \forall\sigma\ \mathcal{O}(R)(\sigma) = M(R)(\gamma)(\sigma)$. As this holds trivially for $\sigma = \perp$, in the sequel assume $\sigma \in \Sigma_0$. We prove Egli-Milner inclusion in both directions.

(1) $\mathcal{O}(R)(\sigma) \sqsubseteq M(R)(\gamma)(\sigma)$ as follows.

Case A. If R and σ are such that $Comp(R)(\sigma) \in P(\Sigma^+)$ then $\mathcal{O}(R)(\sigma) = M(R)(\gamma)(\sigma)$ proof by cases, applying induction on the sum of the lengths of the computation sequences. (Justified by Lemma 16.)

(i) $S \equiv x:=t$

$$\mathcal{O}(\langle E|x:=t\rangle)(\sigma) = \kappa(Comp(\langle E|x:=t\rangle)(\sigma))$$

$$= \kappa(\{\langle\sigma\{\mathcal{V}(t)(\sigma)/x\}\rangle\})$$

$$= \sigma\{\mathcal{V}(t)(\sigma)/x\}$$

$$= M(\langle E|x:=t\rangle)(\gamma)(\sigma)$$

N.B. This result holds for all σ, as $Comp(\langle E|x:=t\rangle)(\sigma) \in P(\Sigma^+)$. By Definition 10 only $\sigma = \perp$ or $S \equiv x:=t$ lead to length $(Comp(R)(\sigma)) = 1$, so the induction basis is provided.

(ii) $S \equiv S_1;S_2$

By Definition 10 and Lemma 16, length $(Comp(\langle E|S_1\rangle)(\sigma)) <$ length $(Comp(\langle E|S_1;S_2\rangle)(\sigma) < \infty$ and length $(Comp(\langle E|S_2\rangle)(\kappa(Comp(\langle E|S_1\rangle)(\sigma)))) <$ length $(Comp(\langle E|S_1;S_2\rangle)(\sigma)) < \infty$. So by induction $\mathcal{O}(\langle E|S_1\rangle)(\sigma) = M(\langle E|S_1\rangle)(\gamma)(\sigma)$ and $\mathcal{O}(\langle E|S_2\rangle)(\kappa(Comp(\langle E|S_1\rangle)(\sigma) = M(\langle E|S_2\rangle)(\gamma)(\kappa(Comp(\langle E|S_1\rangle)(\sigma))$. Consequently,

$$\mathcal{O}(\langle E|S_1;S_2\rangle)(\sigma) = \mathcal{O}(\langle E|S_2\rangle)\circ\mathcal{O}(\langle E|S_1\rangle)(\sigma), \text{ by Lemma 7}$$

$$= \mathcal{O}(\langle E|S_2\rangle)(\kappa(Comp(\langle E|S_1\rangle)(\sigma))), \text{ by Definition 21}$$

$$= M(\langle E|S_2\rangle)(\gamma)(\kappa(Comp(\langle E|S_1\rangle)(\sigma)))$$

$$= M(\langle E|S_2\rangle)(\gamma)(\mathcal{O}(\langle E|S_1\rangle)(\sigma)), \text{ by Definition 21}$$

$$= M(\langle E|S_2\rangle)(\gamma)\circ M(\langle E|S_1\rangle)(\gamma)(\sigma)$$

$$= M(\langle E|S_1;S_2\rangle)(\gamma)(\sigma), \text{ by Definition 31}$$

Cases (iii), (iv) and (v) of Definition 10 can be treated analogously to (ii), applying Lemma 13 when treating Case 5.

Case B. If R and σ are such that $\perp \in Comp(R)(\sigma)$ then $\sigma' \neq \perp$, $\sigma' \in$ $\in O(\langle E|S\rangle)(\sigma)$ implies $\sigma' \in M(\langle E|S\rangle)(\gamma)(\sigma)$, proof by cases, applying induction on the length of computation sequence corresponding to that outcome. There may be more than one sequence satisfying this requirement; in that case choose one arbitrary. We again distinguish the following cases.

(i) $S \equiv x:=t$

Immediately by the above proved equivalence $O(\langle E|x:=t\rangle)(\sigma) = M(\langle E|x:=t\rangle)(\gamma)(\sigma)$. By Definition 10 this is the only case pertaining to length $(\rho) = 1$, $\rho \in Comp(\langle E|S\rangle)(\sigma)$ so the induction basis is provided.

(ii) $S \equiv S_1;S_2$

Consider a computation sequence $\langle\sigma_1(=\sigma),\sigma_2,\ldots,\sigma_n(=\sigma')\rangle \in Comp(\langle E|S_1;S_2\rangle)(\sigma)$. By Definition 10 there is an intermediate state $\sigma_j \neq \perp$ in this sequence such that $\langle\sigma_2,\ldots,\sigma_j\rangle \in Comp(\langle E|S_1\rangle)(\sigma)$ and $\langle\sigma_j,\ldots,\sigma_n\rangle \in Comp(\langle E|S_2\rangle)(\sigma_j)$. As length $(\langle\sigma_2,\ldots,\sigma_j\rangle) <$ length$(\langle\sigma_1,\ldots,\sigma_n\rangle)$ and length$(\langle\sigma_j,\ldots,\sigma_n\rangle) <$ length $(\langle\sigma_1,\ldots,\sigma_n\rangle)$ by induction $\sigma_j \in M(\langle E|S_1\rangle)(\gamma)(\sigma)$ and $\sigma' \in M(\langle E|S_2\rangle)(\gamma)(\sigma_j)$. Consequently, by Definition 31 $\sigma' \in M(\langle E|S_1;S_2\rangle)(\gamma)(\sigma)$.

Cases (iii), (iv) and (v) of Definition 10 can be treated analogeously to (ii), applying Lemma 13 when treating Case 5.

Now combining A and B yields $\forall R\ \forall\gamma\ \forall\sigma\ O(R)(\sigma) \sqsubseteq M(R)(\gamma)(\sigma)$.

(2) Conversely, we prove $M(R)(\gamma) \sqsubseteq O(R)$ as follows:

By Definition 31, it is equivalent to show $N(S)(\gamma\{\psi_i/P_i\}_{i=1}^n) \sqsubseteq O(\langle E|S\rangle)$.

By Definition 31 and Lemma 10, ψ_i can be defined as follows. Let

$\langle\psi_1^0,\ldots,\psi_n^0\rangle = \langle\lambda\sigma\cdot\perp,\ldots,\lambda\sigma\cdot\perp\rangle$

$$\langle\psi_1^{k+1},\ldots,\psi_n^{k+1}\rangle = \langle\Psi_1(\langle\psi_1^k,\ldots,\psi_n^k\rangle),\ldots,\Psi_n(\langle\psi_1^k,\ldots,\psi_n^k\rangle)\rangle,$$

$$k = 0,1,\ldots$$

then $\psi_i = \sqcup_{k=0}^{\infty} \psi_i^k$, $i = 1,\ldots,n$.

By Lemma 11, $N(S)(\gamma\{\psi_i^k/P_i\}_{i=1}^n) = \sqcup_{k=0}^{\infty} N(S)(\gamma\{\psi_i^k/P_i\}_{i=1}^n)$. Therefore it is sufficient to show that for all k, $N(S)(\gamma\{\psi_i^k/P_i\}_{i=1}^n) \sqsubseteq O(\langle E|S\rangle)$. We

apply induction on $\langle k,\ell(S)\rangle$, where $\ell(S)$ is the length of S, i.e. the number of symbols of S with ordering $\langle k_1,\ell_1\rangle < \langle k_2,\ell_2\rangle$ iff $k_1 < k_2$ or $k_1 = k_2$ and $\ell_1 < \ell_2$.

$$N(S)(\gamma\{\psi_i^0/P_i\}_{i=1}^n) = N(S)(\gamma\{\lambda\sigma\cdot\perp/P\}_{i=1}^n) \sqsubseteq \mathcal{O}(\langle E|S\rangle),$$

so the induction basis is satisfied.
We again distinguish the following cases:
(i) $S \equiv x:=t$

$$N(x:=t)(\gamma\{\psi_i^k/P_i\}_{i=1}^n) = \lambda\sigma\cdot\sigma\{V(t)(\sigma)/x\} = \mathcal{O}(\langle E|x:=t\rangle)$$

(ii) $S \equiv S_1;S_2$
$\ell(S_j) < \ell(S_1;S_2)$, so $\langle k,\ell(S_j)\rangle < \langle k,\ell(S_1;S_2)\rangle$, $j = 1,2$.
So by induction $N(S_j)(\gamma\{\psi_i^k/P_i^k\}_{i=1}^n) \sqsubseteq \mathcal{O}(\langle E|S_j\rangle)$, $j = 1,2$.
Consequently, by Lemma 9, 12 and Definition 31,

$$N(S_1;S_2)(\gamma\{\psi_i^k/P_i\}_{i=1}^n) \sqsubseteq \mathcal{O}(\langle E|S_1;S_2\rangle)$$

Cases (iii) and (iv) of Definition 10 can be treated analogeously to (ii)
(v) $S \equiv P$.
By Definition 10, $P \equiv P_j$, $P_j \Leftarrow S_j$ in E. By Lemma 10, $\mathcal{O}(\langle E|P\rangle) = \mathcal{O}(\langle E|S_j\rangle)$
If $k = 0$ there is nothing to prove. Otherwise

$$\begin{aligned} N(S_j)(\gamma\{\psi_i^k/P_i^k\}_{i=1}^n &= \psi_j^k = \Psi_j(\psi_1^{k-1},\ldots,\psi_n^{k-1}) \\ &= N(S_j)(\gamma\{\psi_i^{k-1}/P_i\}_{i=1}^n), \text{ by Definition 31} \\ &\sqsubseteq \mathcal{O}(\langle E|S_j\rangle) \\ &= \mathcal{O}(\langle E|P\rangle) \end{aligned}$$

Combining these results yields $M(R)(\gamma) \sqsubseteq \mathcal{O}(R)$, i.e.
$\forall k\ \forall\gamma\ \forall\sigma\ M(R)(\gamma)(\sigma) \sqsubseteq \mathcal{O}(R)(\gamma)(\sigma)$. □

ACKNOWLEDGEMENTS

I wish to thank Jaco de Bakker for his stimulating remarks and Arie de Bruin for his fruitful cooperation during the process of writing this paper. To Linda Brown and Susan Carolan I am grateful for the efficient typing of the manuscript. The referees I thank for their constructive remarks.

REFERENCES

[1] APT, K.R. & G. PLOTKIN, *A Cook's tour of countable nondeterminism.* To appear in Proc. of the 8 Int. Coll. on Automata, Languages and Programming, Springer (1981).

[2] ARNOLD, A & M. NIVAT, *Metric interpretations of infinite trees and semantics of nondeterministic recursive programs,* Theoretical Computer Science 11 (pp. 108-205) (1980).

[3] BACK, R.-J., *Semantics of unbounded nondeterminism,* in: Proc. 7th Int. Coll. on Automata, Languages and Programming, (J.W. de Bakker and J. van Leeuwen, eds.) pp. 51-63. Lecture Notes in Computer Science 85, Springer (1980).

[4] BAKKER, J.W. de, *Semantics of infinite processes using generalized trees,* in: Mathematical Foundations of Computer Science 1977 (J. Grusk, ed.) Lecture Notes in Computer Science 85, Springer (1977).

[5] BAKKER, J.W. de, *Mathematical theory of program correctness,* Prentice Hall Int. (1980).

[6] BRUIN, A. de, *On the existence of Cook semantics,* Mathematical Centre Report, IW 163/81 (1981).

[7] COOK, S.A., *Soundness and completeness of an axiom system for program verification,* SIAM J. on Computing, 7, pp. 70-90 (1978).

[8] DUGUNDJI, J., *Topology,* Allyn and Bacon, Boston (1966).

[9] DIJKSTRA, E.W., *Guarded commands, nondeterminacy and formal derivations of programs,* Communications ACM 18, pp. 453-457 (1975).

[10] EGLI, H., *A mathematical model for nondeterministic computations*, ETH, Zürich, 1975.

[11] EMERSON, E.A. & E.M. CLARKE, *Characterizing correctness properties of parallel programs using fixed points*, in: Proc. 7th. Int. Coll. on Automata, Languages and Programming (J.W. de Bakker and J. van Leeuwen, eds.) pp. 169-181. Lecture Notes in Computer Science 85, Springer (1980).

[12] HOARE, C.A.R., *Communicating sequential processes*, Comm. ACM 21, 8 (1978).

[13] NIVAT, M., *Infinite words, infinite trees, infinite computations*, in: Foundations of computer science III, part 2 (J.W. de Bakker and J. van Leeuwen, eds), Mathematical Centre Tract 109, pp. 1-52 (1979).

[14] PARK, D., *On the semantics of fair parallellism*, Proc. 1979 Copenhagen Winter School (D. Bjørner, ed.). Lecture Notes in Computer Science 86 (1979).

[15] STOY, J., *Denotational semantics*, The Scott-Strachey Approach to Programming Language Theory, MIT Press, 1977.