Metric Predicate Transformers: Towards a Notion of Refinement for Concurrency

Marcello M. Bonsangue, Erik de Vink
Department of Computer Science, Vrije Universiteit Amsterdam
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
marcello@cs.vu.nl, vink@cs.vu.nl

Joost N. Kok
Department of Computer Science, Utrecht University
P.O. Box 80.089, 3508 TB Utrecht, The Netherlands
joost@cs.ruu.nl

Abstract

For two parallel languages with recursion a compositional weakest precondition semantics is given using two new metric resumption domains. The underlying domains are characterized by domain equations involving functors that deliver 'observable' and 'safety' predicate transformers. Further a refinement relation is defined for this domains and illustrated by rules dealing with concurrent composition. It turns out, by extending the classical duality of predicate vs. state transformers, that the weakest precondition semantics for the parallel languages is isomorphic to the standard metric state transformers semantics. Moreover, the proposed refinement relation on the predicate transformer domain will correspond to the familiar notion of simulation in the state transformer domain.

Contents

1 Introduction

Starting from the seventies —around the publishing of the seminal monograph [Dij76]— a vast amount of research has been undertaken with the point of view of statements as predicate transformers as fundamental. One of the focuses within the field concerns the development of refinement calculi for program derivation, where the soundness of the refinement rules has to be measured with respect to a weakest precondition semantics. Cf., for example, [Bac80, Mor87, GM91]. Parallelism, in the framework of program refinement, is usually treated UNITY-style: the (fair) interleaved execution of an action system consisting of a fixed number of processes with communication through shared variables ([CM88, BKS88, Ser90, Luk91]).

Some results are established dealing with the superposition of action systems. E.g., [BS92] and [BS94], where additionally procedures are incorporated. In general, however, the underlying weakest precondition semantics is not tuned to a *compositional* treatment of concurrency. For some programming languages this may not be an issue at all. For others, in particular those with dynamical object or process creation, it is. Several authors propose a predicate transformer semantics for concurrency for several purposes. For example, more recently, [Bes89, Lam90, SZ92, Bre93]. Taking all together neither of these references combine compositionality and recursion for an explicit parallel operator. The main contribution of the present paper now is the construction of a predicate transformer semantics which does combine all this. Moreover it is indicated how to conservatively extend the refinement calculus with rules dealing with the parallel construct. Also, by comparison with state transformers —extending the classical duality of predicate transformers vs. state transformers— the refinement relation is characterized as the counterpart of simulation for transition systems.

Let us sketch the principal content of the paper. For a proper presentation we will, in fact, treat four languages of increasing expressibility. These are introduced in Section 3. The various merits of the languages, distinguishing between absence or presence of a parallel operator and between finite or unbounded non-determinacy, are depicted as follows:

	finite	unbounded
sequential	$\mathcal{L}_0$	$\mathcal{L}_1$
concurrent	$\mathcal{L}_2$	$\mathcal{L}_3$

Note that by presence of recursion in our languages, it makes sense to distinguish between $\mathcal{L}_0$, $\mathcal{L}_1$ on the one hand, and $\mathcal{L}_2$, $\mathcal{L}_3$ on the other. Technically, the semantics for the parallel $\mathcal{L}_2$, $\mathcal{L}_3$ will be couched in a branching domain which cater for interleaving points, whereas the sequential $\mathcal{L}_0$, $\mathcal{L}_1$ are dealt with in a traditional input/output setting. Below we will focus on the treatment of the parallel language $\mathcal{L}_2$. The model for the language $\mathcal{L}_3$ does not so well support compositional refinement as is the case for $\mathcal{L}_2$. The main ideas can be explained more clearly using the latter language as a vehicle, whereas the discussion of the former bears the load of minor mathematical complications.

In Section 4, the domains Π_2 and Π_3 for $\mathcal{L}_2$ and $\mathcal{L}_3$, respectively, are metric domains. The domains Π_2 and Π_3 can be obtained as unique fixed points of locally contractive functors in the category of complete metric spaces following [AR89, RT93]. Central for both domains is the introduction of a so-called *backward* distance, which enables us to deal with predicate transformers in a metric framework. The domain Π_2 encounters *opens*, whereas the domain Π_3 will deal with *closed* sets. This suggests, following [Smy83, Vic89, Kwi91, BJK95], the suitability of the former for observable or liveness properties in contrast to the latter which might very well be used for safety properties.

The weakest precondition semantics for the languages are given in Section 5. The modus of finite or unbounded nondeterminacy makes a distinction between $\mathcal{L}_0$, $\mathcal{L}_2$ and $\mathcal{L}_1$, $\mathcal{L}_3$. This is reflected in the respective semantics: wp_1 and wp_3 will employ environments; for wp_0 and wp_2 this is not necessary. In particular the model wp_2 for the language $\mathcal{L}_2$ is illustrated with examples involving the parallel operator. It will become clear that the branching structure of the underlying domain Π_2 enforces a more descriptive form of postcondition as one might expect at fore hand. However, at least in the case of $\mathcal{L}_2$, it is pointed out how by what one may call *'saturation'* of predicates one can still reason quite well using wp_2.

For the predicate transformer domains Π_0, Π_1 the isomorphism with domains of state transformers is well known ([Plo79, Bac81, BK94b]). In Section 6 we will extend this duality by showing that Π_2 and Π_3 are isomorphic to metric resumption domains Σ_2 and Σ_3 introduced in [BZ82]. The main purpose of establishing such a relationship is to explain, for $\mathcal{L}_2$, the correspondence of *refinement* in predicate transformer semantics and *simulation* in state transformer semantics. Also, with respect to $\mathcal{L}_2$, some rules are proposed to deal with the parallel operator within the refinement calculus. The situation for $\mathcal{L}_3$ is less satisfactory since, induced by the presence of closure operators, we do not obtain that refinement is a congruence. This issue is, amongst others, discussed in the closing section of concluding remarks and future work.

2 Mathematical Preliminaries

In this section we provide some basic notions and facts on topological and metric spaces. For a more detailed discussion consult [Dug66] or [Eng89].

A *topology* $\mathcal{O}(X)$ on a set X is a collection of subsets of X that is closed under finite intersections and arbitrary unions. Every topology $\mathcal{O}(X)$ is ordered by subset inclusion. The pair $(X, \mathcal{O}(X))$ is called *topological space* and every $o \in \mathcal{O}(X)$ is called an *open set* of the space X. Given an open set $o \in \mathcal{O}(X)$, its complement $c = X \setminus o$ is called a *closed set*. The collection of all closed sets of a topological space $(X, \mathcal{O}(X))$ is denoted by $\mathcal{C}(X)$ and is closed under finite unions and arbitrary intersections. Closed sets are ordered by superset inclusion. For every set $A \subseteq X$ define the closed set $cl(A)$ as the intersection of all closed sets $c \in \mathcal{C}(X)$ such that $A \subseteq c$. A set $q \subseteq X$ is *saturated* if q is the intersection of all open sets $o \in \mathcal{O}(X)$ such that $q \subseteq o$. The collection of the saturated sets is denoted by $\mathcal{Q}(X)$. It is a complete lattice since it is closed under arbitrary intersections.

A topological space is called $\mathcal{T}_0$, if for each pair of distinct points there exists an open set which contains one of the points and that does not contain the other; it is called $\mathcal{T}_1$, if for each pair of distinct points x, y there exists an open set o such that $x \in o$ and $y \notin o$. A topological space $(X, \mathcal{O}(X))$ is *sober* if there is a bijection between X and sets of the form $\{o \in \mathcal{O}(X) \mid x \in o\}$ for every $x \in X$.

A set $K \subseteq X$ is compact (with respect to the topology $\mathcal{O}(X)$) if for every $S \subseteq \mathcal{O}(X)$ such that $K \subseteq \bigcup S$ there exists a finite subset $S' \subseteq S$ such that $K \subseteq \bigcup S'$.

A one-bounded *metric space* is a set X with a function $d : X \times X \to [0,1]$ such that:

(i)	$d(x,y) = d(y,x)$	(iii)	$d(x,x) = 0$
(ii)	$d(x,z) \leq d(x,y) + d(y,z)$	(iv)	$d(x,y) = 0 \Rightarrow x = y$

A set X with a function $d : X \times X \to [0,1]$ satisfying only (i), (ii), and (iii) is called a (one-bounded) *pseudo metric space*.

Let (X, d) be a metric space. A collection of points $(x_i)_{i \in N} \subseteq X$ is called a *Cauchy sequence* if $\forall \epsilon > 0.\ \exists k > 0.\ \forall m, n \geq k.\ d(x_m, x_n) \leq \epsilon$.

A point $x \in X$ is called a *limit* of a Cauchy sequence $(x_i)_{i \in N}$, denoted by $x = lim\ x_i$, if $\forall \epsilon > 0.\ \exists k > 0.\ \forall n \geq k.\ d(x_n, x) \leq \epsilon$. If the limit of a Cauchy sequence exists then it is unique. A metric space X is called *complete* if every Cauchy sequence has a limit.

Let (X, d) be a (pseudo) metric space. The *metric topology* on X is defined by taking as open sets all sets $o \subseteq X$ such that for every $x \in o$ there exists a positive ϵ such that $B_\epsilon(x) \subseteq o$, where $B_\epsilon(x) = \{y \mid d(x,y) < \epsilon\}$. Pseudo metric spaces are, in general, not $\mathcal{T}_0$. Metric spaces, in contrast, are sober and $\mathcal{T}_1$. The closure $cl(A)$ of a set $A \subseteq X$ is the set of all limits of Cauchy sequences in A. It is not hard to see that $A \subseteq cl(A)$, $cl(cl(A)) = A$, $cl(A \cup B) = cl(A) \cup cl(B)$ and that if $A \subseteq B$ then $cl(A) \subseteq cl(B)$. The collection of saturated sets of a metric space (X, d) is $\mathcal{P}(X)$.

A metric space (X, d) is *discrete* if $\forall x \in X.\ \exists \epsilon > 0.\ \forall y \in X.\ d(x,y) < \epsilon$ implies $x = y$. Every set X can be seen as a discrete metric space if endowed with a distance function which assigns to $x, y \in X$, distance 1 if $x \neq y$ and distance 0 otherwise. For a discrete metric space the metric topology of (X, d) coincides with the discrete topology on X (i.e. the topology in which every subset of X is open).

Let (X, d) and (Y, e) be two one-bounded (pseudo) metric spaces. A function $f : (X, d) \to (Y, e)$ is non-expansive if $e(f(x), f(y)) \leq d(x, y)$ for all $x, y \in X$. Complete metric spaces together with non expansive maps form a category, denoted by **CMS**. Of special interest are the *contracting* functions, that is, functions $f:(X,d)\to(Y,e)$ such that

$$\exists 0\leq\epsilon<1.\ \forall x,y\in X.\ e(f(x),f(y))\leq\epsilon\cdot d(x,y).$$

If (X,d) is a complete metric space and $f:X\to X$ is a contracting function then f has a unique fixed point, which is the limit of the Cauchy sequence $(f^n(x))_n$ for an arbitrary $x\in X$ [Ban22].

Let (X,d) and (Y,e) be two one-bounded (pseudo) metric spaces. Define a metric on the disjoint union $X+Y$ by taking the original distance if two points are elements of the same set and distance 1 otherwise. If both X and Y are complete metric spaces so is $X+Y$. Define a metric on the Cartesian product $X\times Y$ by $d(\langle x,y\rangle,\langle x',y'\rangle)=max\{d(x,x'),e(y,y')\}$. Again $X\times Y$ is complete if both X and Y are.

Let (X,d) be a one-bounded (pseudo) metric spaces. We can define a distance between $A,B\in\mathcal{P}(X)$ by

$$d_H(A,B)\quad=\quad max\{\sup\{\inf\{d(a,b)\mid b\in B\}\mid a\in A\},\sup\{\inf\{d(a,b)\mid a\in A\}\mid b\in B\}\}$$

with the convention that $\inf\emptyset=1$ and $\sup\emptyset=0$. The above distance is often called Hausdorff distance. In general $(\mathcal{P}(X),d_H)$ is a pseudo metric space: different subsets of X having the same collection of limit points will have distance 0.

However, the collection of compact subsets of X (denoted by $\mathcal{P}_{co}(X)$) and the collection of closed subsets of X (denoted by $\mathcal{P}_{cl}(X)$) are metric spaces when taken with the Hausdorff distance. Also, if (X,d) is a complete metric space then both $(\mathcal{P}_{co}(M),d_H)$ and $(\mathcal{P}_{cl}(M),d_H)$ are complete metric spaces [Hah48, Kur56]. Below we give two properties of the Hausdorff distance which will be useful later.

2.1. Proposition. *Let (M,d) be a metric space. Then for all $\delta\geq 0$ it holds that*

$$\begin{array}{rll} d_H(A,B)\leq\delta\ \Leftrightarrow & \forall\epsilon>0. & (\forall a\in A.\ \exists b\in B.\ d(a,b)\leq\delta+\epsilon)\\ & \& & (\forall b\in B.\ \exists a\in A.\ d(b,a)\leq\delta+\epsilon)\end{array}$$

for all $A,B\subseteq M$. □

While the above proposition is standard (see [Eng89]) the following one seems to be new.

2.2. Proposition. *Let (M,d) be a metric space and $\mathcal{X},\mathcal{Y}\in\mathcal{P}(\mathcal{P}(M))$ be $\subseteq$-closed, that is, for all $A,B\in\mathcal{P}(M)$ if $A\in\mathcal{X}$ and $A\subseteq B$ then also $B\in\mathcal{X}$. If $\bigcap\mathcal{X}\in\mathcal{X}$ and $\bigcap\mathcal{Y}\in\mathcal{Y}$ then*

$$d_H(\mathcal{X},\mathcal{Y})\quad=\quad d_H(\bigcap\mathcal{X},\bigcap\mathcal{Y}),$$

where the Hausdorff distance on the left-hand side is taken in $\mathcal{P}(\mathcal{P}(M))$ and that on the right-hand side is taken in $\mathcal{P}(M)$.

Proof. Put $\mathcal{X}_0=\bigcap\mathcal{X}$ and $\mathcal{Y}_0=\bigcap\mathcal{Y}$. Let also $\delta=d_H(\mathcal{X},\mathcal{Y})$ and $\delta_0=d_H(\mathcal{X}_0,\mathcal{Y}_0)$. We let X,Y range over $\mathcal{X},\mathcal{Y}$, respectively. We claim

$$\forall X.\ \exists Y.\ d_H(X,Y)\leq\delta_0+\epsilon \text{ and } \forall Y.\ \exists X.\ d_H(Y,X)\leq\delta_0+\epsilon \tag{1}$$

for an arbitrary $\epsilon>0$. From the above claim $\delta\leq\delta_0$ follows by Property 2.1. Next we prove the claim. Choose some $\epsilon>0$ and $X\in\mathcal{X}$. Put $Y=\mathcal{Y}_0\cup X$. Since $\mathcal{Y}$ is $\subseteq$-closed and $\mathcal{Y}_0\in\mathcal{Y}$ we have that also $Y\in\mathcal{Y}$. Since $X\subseteq Y$ we clearly have $\forall x\in X.\ \exists y\in Y.\ d_H(x,y)\leq\delta_0+\epsilon$. Indeed, for $y\in Y$, either $y\in\mathcal{Y}_0$ or $y\in X$ by definition. So, either $d(y,x)\leq\delta_0+\epsilon$ for some $x\in\mathcal{X}_0\subseteq X$ (since $d(\mathcal{X}_0,\mathcal{Y}_0)\leq\delta_0+\epsilon$ and $\mathcal{X}_0\subseteq X\in\mathcal{X}$) or $d(y,x)=0\leq\delta_0+\epsilon$ from $x=y\in X$. Hence $\forall x\in X.\ \exists y\in Y.\ d_H(x,y)\leq\delta_0+\epsilon$. Symmetrically we derive $\forall Y.\ \exists X.\ d_H(Y,X)\leq\delta_0+\epsilon$ from which follows the result (1) and conclude $\delta=d_H(\mathcal{X},\mathcal{Y})\leq\delta_0$.

In order to show the converse, i.e. $\delta_0\leq\delta$, we establish

$$\forall x\in\mathcal{X}_0.\ \exists y\in\mathcal{Y}_0.\ d(x,y)\leq\delta+\epsilon \text{ and } \forall y\in\mathcal{Y}_0.\ \exists x\in\mathcal{X}_0.\ d(y,x)\leq\delta+\epsilon \tag{2}$$

for all $\epsilon > 0$. Since $d_H(\mathcal{X}, \mathcal{Y}) = \delta$ and $\mathcal{X}_0 \in \mathcal{X}$, $\mathcal{Y}_0 \in \mathcal{Y}$ we can find $Y \in \mathcal{Y}$ and $X \in \mathcal{X}$ by Property 2.1 such that

$$d_H(\mathcal{X}_0, Y) \leq \delta + \epsilon \quad \text{and} \quad d_H(X, \mathcal{Y}_0) \leq \delta + \epsilon.$$

From this we obtain the result (2), for $\mathcal{Y}_0 \subseteq Y$ and $\mathcal{X}_0 \subseteq X$, respectively. Therefore $\delta_0 = d_H(\mathcal{X}_0, \mathcal{Y}_0) \leq \delta$. □

Finally we consider (multi-)function spaces. Let (X, d) be a pseudo metric space and S be a set. The function space $S \to X$ can be turned into a pseudo metric space by defining for every $f, g \in S \to X$ a (forward) distance

$$d_F(f, g) = \sup\{d(f(s), g(s)) \mid s \in S\}. \tag{3}$$

If X is a complete metric space then so is $(S \to X, d_F)$. In case the metric space X is on the left hand side we consider the multi-function space from X to S. Recall that the multi-function (or equivalently, relation) space from X to S is the same of the function space $X \to \mathcal{P}(S)$. It can be turned into a pseudo metric space by defining for every $f, g \in X \to \mathcal{P}(S)$ a (backward) distance

$$d_B(f, g) = sup\{d_H(f^-(s), g^-(s)) \mid s \in S\} \tag{4}$$

where $f^-(s) = \{x \in X \mid s \in f(x)\}$ is the inverse of the relation f. Note that in general X being a complete metric space does not imply that $(X \to \mathcal{P}(S), d_B)$ is a complete metric space. Two different sufficient conditions in order to obtain completeness will be given in the next section.

If X and Y are two complete metric space, then $(X \xrightarrow{1} Y, d_F)$ is also a complete metric space, where $X \xrightarrow{1} Y$ is the collection of all non-expansive functions from X to Y.

In [AR89, RT93], generalizing the results of [BZ82], a method has been developed to obtain complete metric spaces as solutions of domain equations of the form $M \cong F(M)$, where F is a functor from the category **CMS** of complete metric spaces and non-expansive functions to **CMS**. We can ensure that a solution exists by considering *locally contracting functors*, that is, a functor $F : \mathbf{CMS} \to \mathbf{CMS}$ such that for every two complete metric spaces X and Y, the mapping

$$F_{X,Y} : \mathbf{CMS}(X, Y) \to \mathbf{CMS}(F(X), F(Y))$$

is contractive, where $F_{X,Y}(f) = F(f)$ for every $f \in X \xrightarrow{1} Y$. If $F_{X,Y}$ is non-expansive, for all objects X, Y, then the functor is called *locally non-expansive.*

For example, for complete metric spaces X and Y and a set S, the following operations (with associated metrics as defined above) can be extended to non-expansive functors [AR89, RT93]: $S \to X$, $X + Y$, $X \times Y$, $\mathcal{P}_{co}(X)$ and $\mathcal{P}_{cl}(X)$.

We use the contracting functor $\frac{1}{2} \cdot (-)$ which maps a complete metric space (X, d) to the complete metric space (X, e), where $e(x, y) = \frac{1}{2} \cdot d(x, y)$ for all $x, y \in X$. Composition of a non-expansive functor with a contractive one gives a contractive functor.

3 Four Languages with Recursion

In this section we define four languages with recursion. We start from a sequential finite non-deterministic language $\mathcal{L}_0$, then either we allow unbounded non-determinism obtaining the sequential language $\mathcal{L}_1$ or we allow, by adding a parallel operator, non-sequentiality obtaining the finitely non-deterministic language $\mathcal{L}_2$. If we allow both non-sequentiality and unbounded non-determinism we finally obtain the language $\mathcal{L}_3$. The following diagram shows the relationships among the four languages.

$$\begin{tikzcd}
 & \mathcal{L}_3 & \\
\mathcal{L}_1 \arrow[ur] & & \mathcal{L}_2 \arrow[ul] \\
 & \mathcal{L}_0 \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

The languages $\mathcal{L}_0$ and $\mathcal{L}_1$ are variations of languages for which the weakest precondition semantics are well known (see [Dij76, Hes89, BW90, BK94b]). The compositional weakest precondition semantics for the languages $\mathcal{L}_2$ and $\mathcal{L}_3$ —to be presented in Section 5— constitute the main contribution of the present paper.

To define the languages, we need as basic blocks the sets $(v \in) \mathit{IVar}$ of (individual) variables, $(e \in) \mathit{Exp}$ expressions, $(b \in) \mathit{BExp}$ boolean expressions, and $(x \in) \mathit{PVar}$ of procedure variables, respectively. For a fixed set of values Val, the set of states $(s, t \in) \mathcal{S}$ is given by $\mathcal{S} = \mathit{IVar} \to \mathit{Val}$. Also, we postulate valuations

$$\mathcal{E} : \mathit{Exp} \to (\mathcal{S} \to \mathit{Val}) \quad \text{and} \quad \mathcal{B} : \mathit{BExp} \to (\mathcal{S} \to \{tt, ff\}).$$

All the languages below have assignment, conditionals $b\!\to$, sequential composition ';', demonic choice '$\Box$' and recursion through procedure variables. The conditional $b\!\to$ deadlocks in a state in which the boolean expression b does not evaluate to true and act as a skip otherwise. The sequential composition is as usual. The demonic choice executes one of its components. The adjective 'demonic' refers to the game theoretic point of view of angels vs. demons, where the latter will always make those choices that lead the computation in the wrong direction. Finally, the intended meaning of the handling of procedure variables is body replacement.

First we introduce the sequential language $\mathcal{L}_0$ with finite non-determinism.

3.1. Definition. (i) The set $(S \in) \mathit{Stat}_0$ of *statements* is given by
$S \;::=\; v := e \mid b\!\to \mid x \mid [S;S] \mid [S \Box S]\,.$
(ii) The set $(G \in) \mathit{GStat}_0$ of *guarded statements* is given by
$G \;::=\; v := e \mid b\!\to \mid [G;S] \mid [G \Box G]\,.$
(iii) The set $(d \in) \mathit{Decl}_0$ is defined by $\mathit{Decl}_0 = \mathit{PVar} \to \mathit{GStat}_0$.
(iv) The language $\mathcal{L}_0$ is given by $\mathit{Decl}_0 \times \mathit{Stat}_0$.

The rather arbitrary choice for the bracketing of $\mathcal{L}_0$ is motivated by the wish to make the several definitions in Section 5 fit well.

Dijkstra's guarded commands ([Dij76, DS90]) are a subset of $\mathcal{L}_0$. Indeed, following [Hes88b], the guarded command $b\!\to S$ is equivalent in $\mathcal{L}_0$ —in the sense that they have the same meaning— to $[b\!\to ;S]$; the conditional command **if** $b\!\to S$ **fi** corresponds in $\mathcal{L}_0$ to the procedure variable x declared as $d(x) = [[b\!\to ;S] \Box [\neg b\!\to ;x]]$ and the iteration command **do** $b\!\to S$ **od** is equivalent to the procedure variable $x \in \mathit{PVar}$ declared as $d(x) = [[[b\!\to ;S];x] \Box \neg b\!\to]$. Other derived statements are $\mathit{skip} = \mathbf{true}\!\to$, $\mathit{magic} = \mathbf{false}\!\to$, and $\mathit{div} = x$ where $d(x) = [\mathbf{false}\!\to ;x]$, and $\{b\} = x$ where $d(x) = b\!\to \Box [\neg b\!\to ;x]$. (Here, **true** and **false** are the obvious boolean expressions.)

Though resembling in their name, the guarded *statements* in Definition 3.1 and elsewhere in the paper are completely different —both syntactically and as to their intended meaning— from Dijkstra's guarded *commands.* The declarations $d \in \mathit{Decl}_0$ associate a procedure body to each procedure variable x. For technical reasons (obtaining contractive higher-order transformations with semantic mappings as their unique fixed point), we restrict procedure bodies to guarded statements. Essentially in an element G of GStat, every occurrence of a procedure variable is preceded either by an assignment or a boolean guard.

Next we extend $\mathcal{L}_0$ with unbounded nondeterminacy obtaining the language $\mathcal{L}_1$.

3.2. Definition. (i) The class $(S \in) \mathit{Stat}_1$ of *statements* is given by
$S \;::=\; v := e \mid b\!\to \mid x \mid [S;S] \mid [\Box_I \{S_i \mid i \in I\}],$
where I is a non-empty index set.
(ii) The class $(G \in) \mathit{GStat}_1$ of *guarded statements* is given by
$G \;::=\; v := e \mid b\!\to \mid [G;S] \mid [\Box_I \{G_i \mid i \in I\}],$
where I is a non-empty index set.
(iii) The class $(d \in) \mathit{Decl}_1$ is defined by $\mathit{Decl}_1 = \mathit{PVar} \to \mathit{GStat}_1$.
(iv) The language $\mathcal{L}_1$ is given by $\mathit{Decl}_1 \times \mathit{Stat}_1$.

Clearly every statement in $\mathcal{L}_0$ is a statement in $\mathcal{L}_1$. In fact, $\mathcal{L}_1$ coincides with the demonic part of the monotonic predicate transformers presented in [BW90]. Notice, however, that the language $\mathcal{L}_1$ is a proper class since the index I in the construct '$\Box_I$' can be any non-empty set. For the familiar reason of amalgamation of specifications and commands, e.g. for a lattice theoretic representation ([BW90, Wri90]) we do not want to constrain the size of index sets I in $\Box_I\{S_i \mid i \in I\}$. Also, the random assignment $v{:}=?$, which assigns to the program variable v an arbitrary value in *Val*, can be expressed in $\mathcal{L}_1$ as $\Box\{v{:}= e_a \mid a \in \mathit{Val}\}$ where $e_a \in \mathit{Exp}$ is such that $\mathcal{E}(e_a)(s) = a$ for all $s \in S$.

Next we add the parallel composition '$\|$' to the above languages. The parallel composition of two statements executes atomic statements of either the first component or the second one in arbitrary order. This is the usual interleaving point of view. A simple form of atomization is also present in the languages below in order to facilitate synchronization. The language $\mathcal{L}_2$ extends $\mathcal{L}_0$. It has a parallel composition but the finite nondeterminacy remains finite.

3.3. Definition. (i) The set $(A \in) \mathit{AStat}_2$ of *atomic statements* is given by
$A \ ::= \ v := e \mid b{\rightarrow} \mid [A;A] \mid [A \Box A]\,.$
(ii) The set $(S \in) \mathit{Stat}_2$ of *statements* is given by
$S \ ::= \ A \mid x \mid (S;S) \mid (S \Box S) \mid (S \| S)\,.$
(iii) The set $(G \in) \mathit{GStat}_2$ of *guarded statements* is given by
$G \ ::= \ A \mid (G;S) \mid (G \Box G) \mid (G \| G)\,.$
(iv) The set $(d \in) \mathit{Decl}_2$ is defined by $\mathit{Decl}_2 = \mathit{PVar} \to \mathit{GStat}_2$.
(v) The language $\mathcal{L}_2$ is given by $\mathit{Decl}_2 \times \mathit{Stat}_2$.

Atomic statements are sequential non-deterministic statements which always terminate. Their execution may not be interrupted by any other process. In order to distinguish atomic sequential composition and atomic choice on the one hand, with the ordinary sequential composition and choice on the other, two different parenthesis are used: brackets [,] around atomic statements, normal parentheses around elements not in AStat_2. Consider, for example, the statements $S_1 \equiv ([v{:}=1; v{:}= v+1] \| v{:}=0)$ and $S_2 \equiv ((v{:}=1; v{:}= v+1) \| v{:}=0)$. They are not equivalent since for a given initial state s, after the execution of S_1 the value of the variable v is either 0 or 2 while after the execution of S_2 the value of v is one of $0, 1$ or 2. Clearly, the language $\mathcal{L}_0$ can be considered a sub-language of $\mathcal{L}_2$ (after an opportune renaming of the parentheses). Below we feel free to suppress the outer parentheses for the usual sequential composition.

In $\mathcal{L}_2$ synchronization and communication by shared variable can be implemented using, for example, semaphores. Indeed by using the simple atomization of the language we can define such a monitoring mechanism: Let $v \in \mathit{IVar}$ and $P_v, V_v \in \mathit{PVar}$ declared as follows

$$\begin{aligned} d(P_v) &= ([v = 0{\rightarrow}\,; v{:}=1] \Box (v \neq 0{\rightarrow}\,; P_v)) \\ d(V_v) &= v{:}=0 \end{aligned}$$

Now, in the program $(v{:}=1; ((S_1; V_v) \| (P_v; S_2)))$ the statement S_2 is executed only after the execution of the statement S_1; in the program $(v{:}=0; ((P_v; S_1; V_v) \| (P_v; S_2; V_v)))$ the execution of S_1 cannot be interrupted by the execution of S_2 and vice versa.

The final language that we introduce is the language $\mathcal{L}_3$ with parallel composition and unbounded choice.

3.4. Definition. (i) The class $(A \in) \mathit{AStat}_3$ of *atomic statements* is given by
$A \ ::= \ v := e \mid b{\rightarrow} \mid [A;A] \mid [\Box_I\{A_i \mid i \in I\}]\,.$
where I is a non-empty index set.
(ii) The class $(S \in) \mathit{Stat}_3$ of *statements* is given by
$S \ ::= \ A \mid x \mid (S;S) \mid (\Box_I\{S_i \mid i \in I\}) \mid (S \| S),$
where I is a non-empty index set.
(iii) The class $(G \in) \mathit{GStat}_3$ of *guarded statements* is given by
$G \ ::= \ A \mid (G;S) \mid (\Box_I\{G_i \mid i \in I\}) \mid (G \| G),$
where I is a non-empty index set.

(iv) The class $(d \in) \mathit{Decl}_3$ is defined by $\mathit{Decl}_3 = \mathit{PVar} \to \mathit{GStat}_3$.
(v) The language $\mathcal{L}_3$ is given by $\mathit{Decl}_3 \times \mathit{Stat}_3$.

The conditional delay **await** b can be expressed in $\mathcal{L}_2$ or $\mathcal{L}_3$ by a procedure variable x declared as $b\!\to \Box(\neg b\!\to ;x)$: in a state where b is evaluated to false, the computation loops busy waiting for some other process which possibly will change the state. Note that this definition of **await** b coincides with the definition of $\{b\}$ for $\mathcal{L}_0$. However, the intended meaning of the two conditionals $\{b\}$ and **await** b is the same in a state where b evaluates to true. In case b evaluates to false then $\{b\}$ can evolve to any computation, while **await** b will loop until b becomes satisfied. In the setting of $\mathcal{L}_0$ the two commands will be identified. In the more detailed process description of $\mathcal{L}_3$ (and $\mathcal{L}_2$) they will obtain different meanings.

4 Domains for Predicate Transformers

In this section we introduce four domains Π_1 to Π_3 of predicate transformers for the languages given in Section 3. The domains Π_0 and Π_1 are for the semantics of the languages without parallel composition, Π_0 for bounded nondeterminism and Π_1 for unbounded nondeterminism. The domains Π_2 and Π_3 are for languages with parallel composition, Π_2 for bounded nondeterminism and Π_3 for unbounded nondeterminism.

4.1. Definition. Let S and M be two sets. A *predicate transformer* over M and S is a function $\pi \in \mathcal{P}(M) \to \mathcal{P}(\mathcal{S})$.
(i) A predicate transformer π is called *conjunctive* if for all non-empty $\mathcal{X} \subseteq \mathcal{P}(M)$
$\pi(\bigcap \mathcal{X}) \;=\; \bigcap\{\pi(Q) \mid Q \in \mathcal{X}\}$;
(ii) a predicate transformer π is called *continuous*, if for all $\subseteq$-directed $\mathcal{D} \subseteq \mathcal{P}(M)$
$\pi(\bigcup \mathcal{D}) \;=\; \bigcup\{\pi(Q) \mid Q \in \mathcal{D}\}$.
(Recall that a set $\mathcal{D} \subseteq \mathcal{P}(M)$ is $\subseteq$-directed if it is non-empty and $\forall A, B \in \mathcal{D}.\ \exists C \in \mathcal{D}.\ A, B \subseteq C$.)
(iii) Define Π_0 as the domain of all conjunctive and continuous predicate transformers in $\mathcal{P}(\mathcal{S}) \to \mathcal{P}(\mathcal{S})$, and
(iv) defineΠ_1 as the domain of all conjunctive predicate transformers in $\mathcal{P}(\mathcal{S}) \to \mathcal{P}(\mathcal{S})$.
In the above definition, recall that a set $\mathcal{D} \subseteq \mathcal{P}(M)$ is $\subseteq$-directed if it is non-empty and for all $A, B \in \mathcal{D}$ there exists $C \in \mathcal{D}$ such that $A, B \subseteq C$.

The next step is to define the domain Π_2. This domain is obtained as the solution of a domain equation involving a functor which delivers the so called observable predicate transformers. The backward distance on observable predicate transformers can be characterized in terms of saturated sets. This turn out to be convenient in formulating some properties of the domain Π_2. Several preparatory steps are necessary, before we can give the definition in Definition 4.12.

We use the distance of metric spaces to single out specific predicates (or subsets). Open sets are the observable predicates of M. A set o is called open if for all $x \in o$ there is an $\epsilon > 0$ such that $B_\epsilon(x) = \{y \mid d(x,y) < \epsilon\} \subseteq o$. The intuition of an observable set $o \in \mathcal{O}(M)$ is that we can not distinguish in o by using only finite information a point $x \in o$ from all other points $y \in B_\epsilon(x) \subseteq o$ (where the ϵ depends on the metric on M). Clearly, this intuition is consistent with the topological axioms: open sets are closed under finite intersections and arbitrary unions. (For an extensive treatment of this topological view of observable predicates, see [Smy83, Abr87, Vic89, Kwi91, BK94a].)

In a metric space M, every predicate can be expressed as intersection of metric observable predicates because the topological space $(M, \mathcal{O}(M))$ is $\mathcal{T}_1$. Hence every predicate $P \in \mathcal{P}(M)$ is the intersection (not necessarily finite) of all observable predicates in which P is included (see [BK94a, lemma 3.5]). The same does not holds in general for $\mathcal{T}_0$ spaces (complete partial order, quasi-metric spaces, SFP-domains, etc.) since every predicate which is not upper closed (with respect to the induced topological order) cannot be expressed as intersection of observable predicates.

4.2. Definition. Let S be a set and (M, d) be a metric space. An *observable predicate transformer* over M and S is a function $\pi \in \mathcal{P}(M) \to \mathcal{P}(\mathcal{S})$ such that

(i) it is *multiplicative,* that is, for all $\mathcal{X} \subseteq \mathcal{P}(M)$
$$\pi(\bigcap \mathcal{X}) \quad = \quad \bigcap\{\pi(Q) \mid Q \in \mathcal{X}\};$$
(ii) it is *open-continuous*, that is, for all directed $\mathcal{D} \subseteq \mathcal{O}(M)$
$$\pi(\bigcup \mathcal{D}) \quad = \quad \bigcup\{\pi(Q) \mid Q \in \mathcal{D}\}.$$

We denote the set of all observable predicate transformers over M and S by $OPT(M, \mathcal{S})$.

Observable predicate transformers are determined by the values on the observable predicates $o \in \mathcal{O}(M)$, because for every $P \in \mathcal{P}(M)$ we have by multiplicativity

$$\pi(P) \quad = \quad \pi(\bigcap\{o \in \mathcal{O}(M) \mid P \subseteq o\} = \bigcap\{\pi(o) \mid P \subseteq o\}.$$

For readers familiar with the paper [BJK95] it follows that $OPT(M, \mathcal{S})$ coincides with the set of all M-multiplicative and continuous functions from the observation frame $\mathcal{O}(M) \rhd\!\!\rightarrow \mathcal{P}(M)$ to the observation frame $\mathcal{P}(\mathcal{S}) \rhd\!\!\rightarrow \mathcal{P}(\mathcal{S})$. Also, since the topological space $(M, \mathcal{O}(M))$ is sober, $OPT(M, \mathcal{S})$ coincides with the continuous functions $\pi \in \mathcal{O}(M) \to \mathcal{P}(\mathcal{S})$ such that

$$\pi(\bigcap \mathcal{X}) \quad = \quad \bigcap\{\pi(Q) \mid Q \in \mathcal{X}\}$$

for all finite $\mathcal{X} \subseteq \mathcal{O}(M)$.

We turn $OPT(M, \mathcal{S})$ into a pseudo metric space using the backward distance: for $\pi_1, \pi_2 \in OPT(M, \mathcal{S})$ we have

$$\begin{aligned} d_B(\pi_1, \pi_2) \quad &= \quad \sup\{d_H(\pi_1^{-1}(s), \pi_2^{-1}(s)) \mid s \in \mathcal{S}\} \\ &= \quad \sup\{d_H(\{P \mid s \in \pi_1(P)\}, \{P \mid s \in \pi_2(P)\}) \mid s \in \mathcal{S}\}. \end{aligned}$$

where d_H on the right-hand side denotes the Hausdorff distance for $\mathcal{P}(\mathcal{P}(M))$.

Next we want to prove that if (M, d) is a complete metric space, so is $(OPT(M, \mathcal{S}), d)$. Before proving the above fact, formally presented below as Theorem 4.8, we give an alternative definition of this metric. This alternative definition is based on the following notion of saturation.

4.3. Definition. For $s \in \mathcal{S}$ and $\pi \in OPT(M, \mathcal{S})$ define $Q(s, \pi) \subseteq M$ by

$$Q(s, \pi) \quad = \quad \bigcap\{P \mid s \in \pi(P)\}$$

The set $Q(s, \pi)$ is referred to as the *saturation* of π with respect to s.

4.4. Lemma. *For given $s \in \mathcal{S}$ and $\pi \in OPT(M, \mathcal{S})$, the saturation $Q(s, \pi)$ is the unique subset of M satisfying*

$$s \in \pi(P) \quad \Leftrightarrow \quad Q(s, \pi) \subseteq P,$$

for all $P \in \mathcal{P}(M)$.

Proof. Pick $s \in \mathcal{S}$, $\pi \in OPT(M, \mathcal{S})$. We first check that $Q(s, \pi)$ fulfill the requirements: Let $P \in \mathcal{P}(M)$.

[$\Rightarrow$] Suppose $Q(s, \pi) \subseteq P$, then by definition of $Q(s, \pi)$, $\bigcap\{P' \mid s \in \pi(P')\} \subseteq P$, hence, by multiplicativity of π, $s \in \bigcap\{\pi(P') \mid s \in \pi(P')\} \subseteq \pi(P)$.

[$\Leftarrow$] Direct from the definition of $Q(s, \pi)$.

We next check the uniqueness of $Q(s, \pi)$ in satisfying the requirement of the lemma. Let $Q' \subseteq M$ such that, for all $P \in \mathcal{P}(M)$,

$$Q' \subseteq P \quad \Leftrightarrow \quad s \in \pi(P).$$

Suppose $x \in Q' \setminus Q(s, \pi)$ and consider $P = M \setminus \{x\}$. We have

$$Q(s, \pi) \subseteq P \quad \Leftrightarrow \quad s \in \pi(P) \;\Leftrightarrow\; Q' \subseteq P$$

contradicting $x \in Q'$. Therefore $Q(s, \pi) \subseteq Q'$ and we conclude, on symmetric considerations, $Q(s, \pi) = Q'$. □

4.5. Remark. As consequence of Lemma 4.4 we have that $s \in \pi(Q(s,\pi))$ and hence

$$Q(s,\pi) \quad = \quad \bigcap\{P \mid s \in \pi(P)\} \in \{P \mid s \in \pi(P)\}.$$

Hence, by Theorem 2.2 we have for every $s \in \mathcal{S}$ and $\pi_1, \pi_2 \in OPT(M,\mathcal{S})$

$$d_H(\{P \mid s \in \pi_1(P)\}, \{P \mid s \in \pi_2(P)\}) \quad = \quad d_H(Q(s,\pi_1), Q(s,\pi_2)).$$

where the Hausdorff distance on the left-hand side is taken in $\mathcal{P}(\mathcal{P}(M))$ while that on the right hand-side is taken in $\mathcal{P}(M)$. Therefore we can conclude for every $\pi_1, \pi_2 \in OPT(M,\mathcal{S})$

$$d_B(\pi_1,\pi_2) \quad = \quad \sup\{d_H(Q(s,\pi_1), Q(s,\pi_2)) \mid s \in \mathcal{S}\}.$$

Hence we have an alternative definition for the metric on $OPT(M,\mathcal{S})$.

4.6. Lemma. *For any $s \in \mathcal{S}$ and $\pi \in OPT(M,\mathcal{S})$, the saturation $Q(s,\pi)$ is compact in the topology of M.*

Proof. Let $\mathcal{D} \subseteq \mathcal{O}(M)$ be a directed cover of $Q(s,\pi)$. Then we have $Q(s,\pi) \subseteq \bigcup \mathcal{D}$. Hence, by Lemma 4.4, $s \in \pi(\bigcup \mathcal{D})$. By open continuity of π it follows that $s \in \bigcup\{\pi(O) \mid O \in \mathcal{D}\}$. So, $\exists O \in \mathcal{D}.\ s \in \pi(O)$. Again applying Lemma 4.4 we derive $\exists O \in \mathcal{D}.\ Q(s,\pi) \subseteq O$. $\square$

4.7. Remark. The converse of Lemma 4.6 is also true: if $\pi : \mathcal{P}(M) \to \mathcal{P}(\mathcal{S})$ is a multiplicative predicate transformer such that for all $s \in \mathcal{S}$ the saturations $Q(s,\pi)$ are compact in the topology of M then π is open continuous. Indeed, for every directed set $\mathcal{D} \subseteq \mathcal{O}(M)$ we have

$$\begin{aligned}
& s \in \pi(\textstyle\bigcup \mathcal{D}) \ \Leftrightarrow\ Q(s,\pi) \subseteq \textstyle\bigcup \mathcal{D} & \\
& \Leftrightarrow\ \ \exists o \in \mathcal{D}.\ Q(s,\pi) \subseteq o & \text{[compactness of } Q(s,\pi)] \\
& \Leftrightarrow\ \ \exists o \in \mathcal{D}.\ s \in \pi(o) & \text{[Lemma 4.4,]}
\end{aligned}$$

that is, π is open continuous.

We still due to the reader the proof of completeness of $(OPT(M,\mathcal{S}), d)$. It is given in the following theorem.

4.8. Theorem. *If (M,d) is a complete metric space, so is $(OPT(M,\mathcal{S}), d)$*

Proof. By Remark 4.5 we use the equivalent Hausdorff metric on the saturated sets. We only check the condition

$$d(\pi_1,\pi_2) = 0 \quad \Rightarrow \quad \pi_1 = \pi_2$$

for d being a metric on $OPT(M,\mathcal{S})$ (the other conditions follow from the respective properties of d_H): Suppose $d(\pi_1,\pi_2) = 0$, then $\forall s \in \mathcal{S}.\ Q(s,\pi_1) = Q(s,\pi_2)$. Hence, for all $P \in \mathcal{P}(M)$, by Lemma 4.4

$$s \in \pi_1(P) \quad \Leftrightarrow \quad Q(s,\pi_1) \subseteq P \ \Leftrightarrow\ Q(s,\pi_2) \subseteq P \ \Leftrightarrow\ s \in \pi_2(P).$$

So, $\forall P \in \mathcal{P}(M).\ \pi_1(P) = \pi_2(P)$ and $\pi_1 = \pi_2$.
Next we check completeness of $OPT(M,\mathcal{S})$ with respect to d. Suppose $(\pi_i)_i$ is a Cauchy sequence in $OPT(M,\mathcal{S})$. Then, for each $s \in \mathcal{S}$ the sequence $(Q(s,\pi_i))_i$ is Cauchy in $\mathcal{P}_{co}(M)$ with respect to d_H. Put, for $s \in \mathcal{S}$, $Q_s = \lim_i Q(s,\pi_i)$, then Q_s in compact in M since $\mathcal{P}_{co}(M)$ is a complete metric space [Kur56]. Now define $\pi : \mathcal{P}(M) \to \mathcal{P}(\mathcal{S})$ by

$$\pi(P) \quad = \quad \{s \in \mathcal{S} \mid Q_s \subseteq P\}.$$

We claim $\pi \in OPT(M,\mathcal{S})$ and $\pi = \lim_i \pi_i$.
(multiplicativity) Let $\mathcal{X} \subseteq \mathcal{P}(M)$. We have to verify $\pi(\bigcap \mathcal{X}) = \bigcap\{\pi(P) \mid P \in \mathcal{X}\}$. Indeed, $s \in \pi(\bigcap \mathcal{X})$ if and only if $Q_s \subseteq \bigcap \mathcal{X}$ if and only if $Q_s \subseteq P$ for all $P \in \mathcal{X}$ if and only if $s \in$

$\bigcap\{\pi(P) \mid P \in \mathcal{X}\}$.
(continuity) Let $\mathcal{D} \subseteq \mathcal{O}(M)$ be directed. If $s \in \pi(\bigcup \mathcal{D})$, then $Q_s \subseteq \bigcup \mathcal{D}$ by definition of π. Since the collection of all compact subsets of a complete metric space forms a complete metric space if taken with the Hausdorff distance we have that Q_s is compact. Hence by Remark 4.7, π is open continuous.
($\pi = \lim_i \pi_i$) First note, since $\pi \in OPT(M, \mathcal{S})$, we have by Lemma 4.4 that $Q_s = Q(s, \pi)$. Now pick $\epsilon > 0$ and let $k \in N$ be such that $d(\pi_i, \pi_j) < \epsilon/2$ for $i, j \geq k$. Then we have $d(Q(s, \pi_i), Q(s, \pi_j)) < \epsilon/2$ for $i, j \geq k$. Hence, by continuity of d, for $j \geq k$,

$$\begin{aligned}
&d(Q(s, \pi), Q(s, \pi_j)) \\
&\quad = d(Q_s, Q(s, \pi_j)) \\
&\quad = d(\lim_i Q(s, \pi_i), Q(s, \pi_j)) \\
&\quad = \lim_i d(Q(s, \pi_i), Q(s, \pi_j)) \\
&\quad \leq \epsilon/2.
\end{aligned}$$

Therefore $d_B(\pi, \pi_j) = \sup\{d_H(Q(s, \pi), Q(s, \pi_j)) \mid s \in \mathcal{S}\} < \epsilon$ from which $\pi = \lim_i \pi_i$ follows. □

Next we start preparations for using OPT in domain equations.

4.9. Definition. The functor $OPT(-, \mathcal{S}) : \mathbf{CMS} \to \mathbf{CMS}$, for given set S, on the category **CMS** of complete metric spaces with non-expansive maps is defined as follows:
(i) $OPT(M, \mathcal{S})$ is the set of predicate transformers $OPT(M, \mathcal{S})$ endowed with the backward distance induced by d;
(ii) For $f : M \xrightarrow{1} M'$ non-expansive $OPT(f, \mathcal{S}) = \lambda\pi.\, \pi \circ f^{-1}$, that is,

$$OPT(f, \mathcal{S})(\pi)(P') \quad = \quad \pi(f^{-1}(P'))$$

for $\pi \in OPT(M, \mathcal{S})$ and $P' \in \mathcal{P}(M')$.

Before we prove that the functor $OPT(-, \mathcal{S})$ is well defined we need the following proposition.

4.10. Proposition. *Let $f : M \xrightarrow{1} M'$ be arbitrary and $\pi \in OPT(M, \mathcal{S})$. Suppose $\pi \circ f^{-1} \in OPT(M', \mathcal{S})$. Then it holds that*

$$Q(s, \pi \circ f^{-1}) \quad = \quad f(Q(s, \pi))$$

for any $s \in \mathcal{S}$.

Proof. Using Lemma 4.4 it suffices to check the equivalence

$$f(Q(s, \pi)) \subseteq P' \quad \Leftrightarrow \quad s \in \pi(f^{-1}(P')),$$

for all $P' \in \mathcal{P}(M')$. This can be derived as follows:

$$\begin{aligned}
&f(Q(s, \pi)) \subseteq P' \\
&\Leftrightarrow\ Q(s, \pi) \subseteq f^{-1}(P) \\
&\Leftrightarrow\ s \in \pi(f^{-1}(P')) && \text{[Lemma 4.4 for } \pi]
\end{aligned}$$

where the first equivalence is obtained from the set-theoretic observation $X \subseteq \phi^{-1}(Y) \Leftrightarrow \phi(X) \subseteq Y$, for $\phi : A \to B$ and $X \subseteq A$, $Y \subseteq B$. □

4.11. Lemma. *The functor $OPT(-, \mathcal{S}) : \mathbf{CMS} \to \mathbf{CMS}$ for some fixed set S is well-defined and locally non-expansive.*

Proof. Well-definedness of $OPT(-, \mathcal{S})$ on objects follows from Theorem 4.8. For morphisms it is proved as follows. For a non-expansive function $f : M \to M'$ and $\pi \in OPT(M, \mathcal{S})$ clearly $OPT(f, \mathcal{S})(\pi) : \mathcal{P}(M') \to \mathcal{P}(\mathcal{S})$. Multiplicativity of $OPT(f, \mathcal{S})(\pi)$ directly follows from set-theoretic laws for f^{-1}, while open continuity follows from the metric continuity of f ($f^{-1}(O') \in \mathcal{O}(M)$ for any $O' \in \mathcal{O}(M')$). Hence $OPT(f, \mathcal{S}) : OPT(M, \mathcal{S}) \to OPT(M', \mathcal{S})$ is well-defined. We can observe non-expansiveness of $OPT(f, \mathcal{S})$ as follows: Let $\pi_1, \pi_2 \in OPT(M, \mathcal{S})$. Then

$$
\begin{aligned}
& d_B(OPT(f,\mathcal{S})(\pi_1), OPT(f,\mathcal{S})(\pi_2)) \\
& \quad = \sup\{d_H(Q(s,\pi_1\circ f^{-1}), Q(s,\pi_2\circ f^{-1})) \mid s\in\mathcal{S}\} \\
& \quad = \sup\{d_H(f(Q(s,\pi_1)), f(Q(s,\pi_2)) \mid s\in\mathcal{S}\} && \text{[Proposition 4.10]} \\
& \quad \le \sup\{d_H(Q(s,\pi_1), Q(s,\pi_2) \mid s\in\mathcal{S}\} && [f \text{ non-expansive}] \\
& \quad = d_B(\pi_1,\pi_2).
\end{aligned}
$$

We establish functoriality of $OPT(-,\mathcal{S})$ by

(i) $OPT(id,\mathcal{S}) = id$, for $id: M\to M$ the identity.

(ii) $OPT(g\circ f,\mathcal{S}) = OPT(g,\mathcal{S})\circ OPT(f,\mathcal{S})$ for $f: M \xrightarrow{1} M'$, $g: M' \xrightarrow{1} M''$ is directly verified for all arguments $\pi\in OPT(M,\mathcal{S})$ using the equality $(g\circ f)^{-1} = f^{-1}\circ g^{-1}$.

Finally, we check that the functor $OPT(-,\mathcal{S}): \mathbf{CMS}\to\mathbf{CMS}$ is locally non-expansive. Take $f,g\in\mathbf{CMS}(M,M')$, that is, $f,g: M\to M'$, non-expansive. We have to check

$$
d(OPT(f,\mathcal{S}), OPT(g,\mathcal{S})) \le d(f,g) \tag{5}
$$

with the distance in the left-hand side taken in $OPT(M,\mathcal{S}) \to OPT(M',\mathcal{S})$. So, pick $\pi\in OPT(M,\mathcal{S})$. Then

$$
\begin{aligned}
& d(OPT(f,\mathcal{S})(\pi), OPT(g,\mathcal{S})(\pi)) \\
& \quad = \sup\{d(Q(s, OPT(f,\mathcal{S})(\pi)), Q(s, OPT(g,\mathcal{S})(\pi))) \mid s\in\mathcal{S}\} \\
& \quad = \sup\{d(f(Q(s,\pi)), g(Q(s,\pi))) \mid s\in\mathcal{S}\} \\
& \quad \le d(f,g),
\end{aligned}
$$

since, for any $s\in\mathcal{S}$, $d(f(Q(s,\pi)), g(Q(s,\pi)) \le d(f,g)$. Thus

$$
d(OPT(f,\mathcal{S})(\pi), OPT(g,\mathcal{S})(\pi)) \quad \le \quad d(f,g).
$$

Now (5) follows directly by definition of the distance on $OPT(M,\mathcal{S}) \to OPT(M',\mathcal{S})$. □

Finally we can define the domain Π_2 of metric predicate transformers with resumptions as the unique (up to isometry) solution of a contractive functor.

4.12. Definition. The complete metric space $(\pi\in)\,\Pi_2$ is given by

$$
\Pi_2 \quad\cong\quad OPT(\mathcal{S} + \mathcal{S}\times\Pi_2/2, \mathcal{S})
$$

for given set of states S.

Recall that $OPT(\mathcal{S}+\mathcal{S}\times\Pi_2/2,\mathcal{S})$ are all the multiplicative and open-continuous functions in $\mathcal{P}(\mathcal{S}+\mathcal{S}\times\Pi_2/2)\to\mathcal{P}(\mathcal{S})$.

The intuition behind an element $\pi\in\Pi_2$ is as follows. Given a set P it yields the set of all states such that when execution starts in one of the states, it either terminates in a $s\in P$, or it makes a first (atomic) step to a state s, with $\langle s,\pi'\rangle\in P$, passes an interleaving point, and executes further according to the predicate transformer π'. We have not yet made a formal semantic mapping from syntax to the domains, but it might help the reader to consider the following example. Consider the statement $x := 1;y := 2$. The corresponding metric predicate transformer π is

$$
\pi(P) \quad = \quad \{s \mid \langle s\{1/x\}, \pi'\rangle \in P\}
$$

with π' (the semantics of $y := 2$) such that

$$
\pi'(P) \quad = \quad \{s \mid s\{2/y\}\in P\}.
$$

If we consider the atomic version $[x := 1;y := 2]$ (in which the interleaving point is not present) then we have as corresponding predicate transformer

$$
\rho(P) \quad = \quad \{s \mid s\{1/x\}\in\pi'(P)\} = \{s \mid s\{1/x, 2/y\}\in P\}.
$$

We saw from the above discussion on observable predicate transformers, that a topology on a set M can be used to distinguish different kinds of predicates. Closed sets in $\mathcal{C}(M)$ are complements of open sets, and are predicates for which we can tell whether the predicate does not hold for a point x on the basis of a finite amount of information. They are usually called safety predicates ("something bad does not happen"). We refer to predicate transformers characterized on the closed sets as safety predicate transformers. We shall define our last domain Π_3 using a functor for safety predicate transformers. Based on our efforts done for Π_2 already, this is now straightforward.

4.13. Definition. Let S be a set and (M, d) be a metric space. A *safety predicate transformer* over M and S is a multiplicative function $\pi \in \mathcal{P}(M) \to \mathcal{P}(\mathcal{S})$ such that $\pi(P) = \pi(cl(P))$ for all $P \in \mathcal{P}(M)$. We denote the set of all safety predicate transformers over M and S by $SPT(M, \mathcal{S})$

A safety predicate transformer does not distinguish a predicate P from its closure (with respect to the metric on M), therefore they are determined uniquely by their value in the closed sets. Notice that $SPT(M, \mathcal{S})$ is well defined because for every $\mathcal{X} \subseteq \mathcal{P}(M)$ we have

$$\begin{aligned}
&\pi(\bigcap \mathcal{X}) \\
&= \bigcap\{\pi(P) \mid P \in \mathcal{X}\} \\
&= \bigcap\{\pi(cl(P)) \mid P \in \mathcal{X}\} \\
&= \pi(\bigcap\{cl(P) \mid P \in \mathcal{X}\}) \\
&= \pi(cl(\bigcap \mathcal{X})).
\end{aligned}$$

We can turn $SPT(M, \mathcal{S})$ in a metric space using the backward distance (already employed for the construction of the metric space Π_2). For $\pi_1, \pi_2 \in SPT(M, \mathcal{S})$ we have

$$\begin{aligned}
&d(\pi_1, \pi_2) \\
&= \sup\{d_H(\pi_1^-(s), \pi_2^-(s)) \mid s \in \mathcal{S}\} \\
&= \sup\{d_H(\{P \mid s \in \pi_1(P)\}, \{P \mid s \in \pi_2(P)\}) \mid s \in \mathcal{S}\} \\
&= \sup\{d_H(\{c \in \mathcal{C}(M) \mid s \in \pi_1(c)\}, \{c \in \mathcal{C}(M) \mid s \in \pi_2(c)\}) \mid s \in \mathcal{S}\}
\end{aligned}$$

where d_H on the right-hand side denotes the Hausdorff distance for $\mathcal{P}(\mathcal{C}(M))$. It is not hard to see that $SPT(M, \mathcal{S})$ is a complete metric space if M is a complete metric space and that we can turn $SPT(M, \mathcal{S})$ in a locally non-expansive functor $SPT(-, \mathcal{S}) : \mathbf{CMS} \to \mathbf{CMS}$. (Note that the saturation of $\pi \in SPT(M, \mathcal{S})$ with respect to $s \in \mathcal{S}$, denoted by $Q(s, \pi) = \bigcap\{P \in \mathcal{C}(M) \mid s \in \pi(P)\}$ is not compact but closed.)

We define the domain Π_3 of metric predicate transformers with resumptions to handle unbounded non-determinism together with sequential composition and merge.

4.14. Definition. The complete metric space $(\pi \in) \Pi_3$ is the solution of the equation

$$X \quad = \quad SPT(\mathcal{S} + (\mathcal{S} \times \tfrac{1}{2}(X)), \mathcal{S})$$

for given set of states $\mathcal{S}$.

The resumption domains Π_2 (and Π_3) allow a refinement relation, generalizing the familiar notion of refinement in the setting of Π_0 and Π_1. In Section 6 we will, in fact, propose concrete refinement rules based on Definition 4.15. Also, there the proposed refinement relation will be compared with, for transition systems, standard notion of simulation.

4.15. Definition. A refinement relation R is a binary relation on Π_2 such that if $(\pi, \rho) \in R$ then

$$\forall P.\ \pi(P) \quad \subseteq \quad \rho(\{s \in \mathcal{S} \mid s \in P\} \cup \{\langle s, \rho' \rangle \mid \exists \langle s, \pi' \rangle \in P.\ (\pi', \rho') \in R\}).$$

We say "π is refined by ρ", notation $\pi \trianglelefteq \rho$, if there exists a refinement relation R containing (π, ρ).

With abuse of language, we refer to '$\trianglelefteq$' as the refinement relation. The intuition of the formula $\pi \trianglelefteq \rho$ is as follows. The basic idea is that a refinement reduces the (demonic) non-determinism. It might be helpful to the reader that $R \subseteq \Pi_2 \times \Pi_2$ is a refinement relation if and only if $(\pi, \rho) \in R$ implies for all $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$ there exists $P' \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$ such that

(i) $\pi(P) \subseteq \rho(P')$,
(ii) $P \cap \mathcal{S} = P' \cap \mathcal{S}$, and
(iii) $\forall \langle s, \pi' \rangle \in P.\ \exists \langle s, \rho' \rangle \in P'.\ (\pi', \rho') \in R$.

In the case of standard predicate transformers this implies that the number of initial states increases (for a given fixed postcondition, the weakest precondition becomes weaker): for standard predicate transformers in Π_0 and Π_1 we require

$$\pi \trianglelefteq \rho \quad \Leftrightarrow \quad \forall P \in \mathcal{P}(\mathcal{S}).\ \pi(P) \subseteq \rho(P).$$

For example, the predicate transformer corresponding to $(x := 1 \Box x := 2)$ is refined by the predicate transformer corresponding to $(x := 1)$. The question is: can we take the same definition for the domain Π_2?

This does not work, as is shown by the following example. Consider the statements

$$x := 1;(y := 1 \Box y := 2) \quad \text{and} \quad x := 1; y := 1.$$

We want to have that the first statement is refined by the second one. Based on the definitions to be presented in Section 5, we obtain as predicate transformer for $(y := 1 \Box y := 2)$:

$$\pi'(P) \quad = \quad \{s \mid s\{1/y\} \in P \ \&\ s\{2/y\} \in P\}.$$

As predicate transformer for $x := 1;(y := 1 \Box y := 2)$ we have: $\pi(P) = \{s \mid \langle s\{1/x\}, \pi' \rangle \in P\}$, for $y := 1$: $\rho'(P) = \{s \mid s\{1/y\} \in P\}$, for $x := 1; y := 1$: $\rho(P) = \{s \mid \langle s\{1/x\}, \rho' \rangle \in P\}$. If we take (for a fixed s) $P = \{\langle s\{1/x\}, \pi' \rangle\}$ then $\pi(P) = \{s\{\alpha/x\} \mid \alpha \in \mathit{Val}\}$ and $\rho(P) = \emptyset$. Hence $\pi(P)$ is not a subset of $\rho(P)$. Now we can identify the problem: we have to add all pairs $\langle s\{1/x\}, \pi'' \rangle$, with π'' a refinement of π', to the argument of ρ, such that ρ can pick $\langle s\{1/x\}, \rho' \rangle$, which explains the definition above.

5 Predicate Transformer Semantics

Preparatory to the semantics of the four languages discussed, we introduce the predicates $P[e/v]$ and $b \rightarrow P$ obtained from a predicate P.

5.1. Definition. The operations '$(\cdot)[e/v]$' and '$b \rightarrow (\cdot)$' are given by

$$\begin{array}{lcl} P[e/v] & = & \{s \in \mathcal{S} \mid s[\alpha/v] \in P, \alpha = \mathcal{E}(e)(s)\} \\ b \rightarrow P & = & \{s \in \mathcal{S} \mid \mathcal{B}(b)(s) \Rightarrow s \in P\}. \end{array}$$

In fact, the definition above is parametric, in that it will vary to range over Π_0 through Π_3. The definitions, however, are conceptually the same, and therefore gathered here.

Further, on Π_0, we have the usual operators ';' and '$\Box$'.

5.2. Definition. The operators ';' and '$\Box$' on Π_0 are given by

$$\begin{array}{lcl} \pi_1;\pi_2(P) & = & \pi_1(\pi_2(P)), \\ (\pi_1 \Box \pi_2)(P) & = & \pi_1(P) \cap \pi_2(P). \end{array}$$

Well-definedness of ';' and '$\Box$' is straightforward. The weakest precondition semantics wp_0 for $\mathcal{L}_0$ can now be given.

5.3. Definition. The weakest precondition semantics $wp_0[\![\cdot]\!] : \mathcal{L}_0 \to \Pi_0$ is the least function in $\mathcal{L}_0 \to \Pi_0$ (with respect to the pointwise extension of the subset order) such that

$$\begin{array}{lcl} wp_0[\![\langle d, v := e\rangle]\!] & = & \lambda P.\, P[e/v], \\ wp_0[\![\langle d, b{\to}\rangle]\!] & = & \lambda P.\, b{\to}\, P, \\ wp_0[\![\langle d, x\rangle]\!] & = & wp_0[\![\langle d, d(x)\rangle]\!], \\ wp_0[\![\langle d, [S_1;S_2]\rangle]\!] & = & wp_0[\![\langle d, S_1\rangle]\!];wp_0[\![\langle d, S_2]\!]\rangle, \\ wp_0[\![\langle d, [S_1\Box S_2]\rangle]\!] & = & wp_0[\![\langle d, S_1\rangle]\!]\Box wp_0[\![\langle d, S_2\rangle]\!]. \end{array}$$

The above semantics for $\mathcal{L}_0$ is well-known (see [Dij76]) including the separate treatment of guards $b{\to}$ ([Hes88a]). Probably less familiar is the justification of the above definition of wp_0. The semantics can be obtained as least fixed point of a higher order transformation, as promoted originally in a metric setting in [KR90], thus avoiding employment of environments. We need at first a map which assigns a natural number to every statement in $\mathcal{L}_0$ to be used in proofs based on induction.

5.4. Definition. The map $wgt_0 : \mathcal{L}_0 \to N$ is given by

$$\begin{array}{lcl} wgt_0(\langle d, v := e\rangle) & = & 1, \\ wgt_0(\langle d, b \to\rangle) & = & 1, \\ wgt_0(\langle d, x\rangle) & = & wgt_0(\langle d, d(x)\rangle) + 1, \\ wgt_0(\langle d, S_1;S_2\rangle) & = & wgt_0(\langle d, S_1\rangle) + 1, \\ wgt_0(\langle d, S_1\Box S_2\rangle) & = & \max\{wgt_0(\langle d, S_1\rangle), wgt_0(\langle d, S_2\rangle)\} + 1, \end{array}$$

The function wgt_0 is well defined for each pair $\langle d, S\rangle \in \mathcal{L}_0$ as can be easily seen by induction on the syntactic complexity first of guarded statements and then of general statements (for more on the use of the weight function see [BV95]). We are now ready for the justification of wp_0. It is based on a mapping $\Psi_0 : Sem_0 \to Sem_0$ where $(F \in)\, Sem_0 = \mathcal{L}_0 \to \Pi_0$. Pivotal is the clause

$$\Psi_0(F)(S_1;S_2) \quad = \quad \Psi_0(F)(S_1);\phi(S_2)$$

for the sequential command. (Note that F and not $\Psi_0(F)$ is applied to the second component S_2.) The least fixed point of this continuous endomorphism will satisfy the conditions of Definition 5.3.

5.5. Lemma. *Let $F \in Sem_0 = \mathcal{L}_0 \to \Pi_0$. The higher-order transformation $\Psi_0 : Sem_0 \to Sem_0$ is given by*

$$\begin{array}{lcl} \Psi_0(F)(\langle d, v := e\rangle) & = & \lambda P.\, P[e/v], \\ \Psi_0(F)(\langle d, b \to\rangle) & = & \lambda P.\, b{\to}\, P, \\ \Psi_0(F)(\langle d, x\rangle) & = & \Psi_0(F)(\langle d, d(x)\rangle), \\ \Psi_0(F)(\langle d, S_1;S_2\rangle) & = & \Psi_0(F)(\langle d, S_1\rangle);F(\langle d, S_2\rangle), \\ \Psi_0(F)(\langle d, S_1\Box S_2\rangle) & = & \Psi_0(F)(\langle d, S_1\rangle)\Box\Psi_0(F)(\langle d, S_2\rangle), \end{array}$$

Then Ψ_0 is well defined, continuous and the function $wp_0[\![\cdot]\!]$ defined in Definition 5.3 is the least fixed point of Ψ_0.

Proof. Well-definedness of Ψ_0 is readily checked. We show the continuity of Ψ_0 for every directed set $\mathcal{D} \subseteq Sem_0$ by induction on $wgt_0(\langle d, S\rangle)$, that is we prove

$$\bigcup_{F\in\mathcal{D}} \Psi_0(F)(\langle d, S\rangle)(P) \quad = \quad \Psi_0(\bigsqcup \mathcal{D})(\langle d, S\rangle)(P)$$

for all statement $\langle d, S\rangle \in \mathcal{L}_0$ and $P \in \mathcal{P}(S)$. Since basic cases are trivial we expand two typical sub-cases of the induction step.

$[x]$

$$\begin{array}{ll} \Psi_0(\bigsqcup \mathcal{D})(\langle d, x\rangle)(P) & \\ \quad = \Psi_0(\bigsqcup \mathcal{D})(\langle d, d(x)\rangle)(P) & \\ \quad = \bigcup_{F\in\mathcal{D}} \Psi_0(F)(\langle d, d(x)\rangle)(P) & \text{[induction hypothesis]} \end{array}$$

$$
\begin{array}{lll}
 & = \bigcup_{F\in\mathcal{D}} \Psi_0(F)(\langle d, x\rangle)(P) & \\
[S_1;S_2] & \quad \Psi_0(\bigsqcup \mathcal{D})(\langle d, S_1;S_2\rangle)(P) & \\
 & = \Psi_0(\bigsqcup \mathcal{D})(\langle d, S_1\rangle);(\bigsqcup \mathcal{D})(\langle d, S_2\rangle)(P) & \\
 & = \Psi_0(\bigsqcup \mathcal{D})(\langle d, S_1\rangle)(\bigcup_{F\in\mathcal{D}} F(\langle d, S_2\rangle)(P)) & \\
 & = \bigcup_{F\in\mathcal{D}} \Psi_0(\bigsqcup \mathcal{D})(\langle d, S_1\rangle)(F(\langle d, S_2\rangle)(P)) & \text{[continuity]} \\
 & = \bigcup_{F\in\mathcal{D}} \bigcup_{F\in\mathcal{D}} \Psi_0(F)(\langle d, S_1\rangle)(F(\langle d, S_2\rangle)(P)) & \text{[induction hypothesis]} \\
 & = \bigcup_{F\in\mathcal{D}} \Psi_0(F)(\langle d, S_1\rangle);F(\langle d, S_2\rangle)(P) & \\
 & = \bigcup_{F\in\mathcal{D}} \Psi_0(F)(\langle d, S_1;S_2\rangle)(P). & \square
\end{array}
$$

Next we give a weakest precondition semantics to the language $\mathcal{L}_1$ of Definition 3.2. Since the language $\mathcal{L}_1$ is a proper class we cannot apply the technique of the higher order transformation as we have done for the previous language.This because $Sem_1 = \mathcal{L}_1 \to \Pi_0$ is a proper class and hence the higher-order transformation $\Psi_1 : Sem_1 \to Sem_1$ even if monotone will not ensure us the existence of least fixed point. This problem can be avoided by using another technique which acts more locally than the higher-order function: the environments. Such an approach to a predicate transformer semantics for a language with recursion and unbounded nondeterminism has been used for example in [BK94b].

5.6. Definition. Let I be a non-empty index set. The operators ';' and '$\Box_I$' on Π_1 are given by

$$
\begin{array}{lll}
\pi_1;\pi_2(P) & = & \pi_1(\pi_2(P)) \\
\Box_I\{\pi_i \mid i \in I\}(P) & = & \bigcap\{\pi_i(P) \mid i \in I\}.
\end{array}
$$

The definition of the $\mathcal{L}_1$-semantics wp_1, which turns out to be more involved then Definition 5.3 can now be given.

5.7. Definition. Put $(\eta \in) \, Env_1 = PVar \to \Pi_1$.

(i) The mapping $wp_1 : Stat_1 \to Env_1 \to \Pi_1$ is given by

$$
\begin{array}{lll}
wp_1(v := e)(\eta) & = & \lambda P.\, P[e/v], \\
wp_1(b\!\to)(\eta) & = & \lambda P.\, b\!\to P, \\
wp_1(x)(\eta) & = & \eta(x), \\
wp_1([S_1;S_2])(\eta) & = & wp_1(S_1)(\eta);wp_1(S_2)(\eta), \\
wp_1([\Box_I\{S_i \mid i \in I\}])(\eta) & = & \Box_I\{wp_1(S_i)(\eta) \mid i \in I\}.
\end{array}
$$

(ii) For all $d \in Decl_1$ the transformation $\mathrm{H}_d : Env_1 \to Env_1$ is given by
$\mathrm{H}_d(\eta)(x) \quad = \quad wp_1(d(x))(\eta)$.

(iii) The weakest precondition semantics $wp_1 : \mathcal{L}_1 \to \Pi_1$ is given by
$wp_1[\![\langle d, S\rangle]\!] \quad = \quad wp_1(S)(\eta_d)$,
where η_d is the least fixed point of H_d.

Monotonicity of H_d can be straightforwardly checked. It is ultimately based on the monotonicity of ';' and '$\Box_I$'. Since Π_1 is a complete partial order, we have that also Env_1 is a complete partial order. Therefore for every declaration $d \in Decl_1$ the function H_d has least fixed point [HP72]. The technique is adapted from [Vin90]. Note that this way – environments as place holders for temporary meanings of procedure variables – intricate problems induced by the cardinality of $\mathcal{L}_1$ are avoided.

5.8. Lemma. *The weakest precondition semantics $wp_1[\![\cdot]\!] : \mathcal{L}_1 \to \Pi_1$ defined in Definition 5.7 is the least function in $Sem_1 = \mathcal{L}_1 \to \Pi_1$ (with respect to the pointwise extension of the subset order) such that*

$$
\begin{array}{lll}
wp_1[\![\langle d, v := e\rangle]\!] & = & \lambda P.\, P[e/v], \\
wp_1[\![\langle d, b\!\to\rangle]\!] & = & \lambda P.\, b\!\to P, \\
wp_1[\![\langle d, x\rangle]\!](P) & = & wp_1[\![\langle d, d(x)\rangle]\!], \\
wp_1[\![\langle d, [S_1;S_2]\rangle]\!] & = & wp_1[\![\langle d, S_1\rangle]\!];wp_1[\![\langle d, S_2]\!]\rangle, \\
wp_1[\![\langle d, \Box_I\{S_i \mid i \in I\}\rangle]\!] & = & (\Box_I\{wp_1[\![\langle d, S_i\rangle]\!] \mid i \in I\}).
\end{array}
$$

Proof. First of all we see that $wp_1[\![\cdot]\!]$ satisfies the requirement above by structural induction on the syntactic complexly of the statements. We treat the only non-trivial case of procedure variables.

$$
\begin{aligned}
& wp_1[\![\langle d, x\rangle]\!] \\
&= wp_1(\langle d, x\rangle)(\eta_d) && \text{[Definition 5.7]} \\
&= \eta_d(x) \\
&= \mathrm{H}_d(\eta_d)(x) && \text{[fixed point property]} \\
&= wp_1(\langle d, d(x)\rangle)(\eta_d) \\
&= wp_1[\![\langle d, d(x)\rangle]\!].
\end{aligned}
$$

Let now $F \in Sem_1$ be another semantics such that satisfies the requirement above. We prove by structural induction on the syntactic complexity of $S \in Stat_1$ that for every $\langle d, S\rangle \in \mathcal{L}_1$

$$wp_1[\![\langle d, S\rangle]\!](P) \quad \subseteq \quad F(\langle d, S\rangle)(P).$$

We consider one sub-case only, namely $S = x \in PVar_1$. Define, for a fixed declaration $d \in Decl_1$, the environment $\eta \in Env_1$ by $\eta(x) = F(\langle d, x\rangle)$. It is not hard to see by structural induction on the syntactic complexity of $S \in Stat_1$ that $wp_1(\langle d, S\rangle)(\eta) = F(\langle d, S\rangle)$. But then we have that η is a fixed point of H_d. Indeed for every $x \in PVar$ we have by Definition 5.7

$$\mathrm{H}_d(\eta)(x) = wp_1(\langle d, d(x)\rangle)(\eta) = F(\langle d, d(x)\rangle) = F(\langle d, x\rangle) = \eta.$$

Since η_d is the least fixed point of H_d we have $\eta_d(x)(P) \subseteq \eta(x)(P)$ for every $x \in PVar$ and $P \in \mathcal{P}(S)$. Finally we have

$$
\begin{aligned}
& wp_1[\![\langle d, x\rangle]\!](P) \\
&= wp_1(\langle d, x\rangle)(\eta_d)(P) \\
&= \eta_d(x)(P) \\
&\subseteq \eta(x)(P) \\
&= wp_1(\langle d, x\rangle)(\eta)(P) \\
&= F(\langle d, x)(P).
\end{aligned}
$$

□

We next turn to the predicate transformer semantics for $\mathcal{L}_2$. As $\mathcal{L}_2$ can be considered an extension of $\mathcal{L}_0$ we will base our present semantics on wp_0. However, the presence of the parallel operator '$\|$' in $\mathcal{L}_2$, invokes, when insisting on a compositional treatment, a more involved domain than Π_0. We will employ the branching domain Π_2. In this domain interleaving points are explicitly represented. This complicates the definitions of the operators on predicate transformers in Π_2 which reflect the constructions available in $\mathcal{L}_2$.

5.9. Definition. The operators ';', '+' and '$\|$' on Π_2 are given by

$$
\begin{aligned}
(\pi_1;\pi_2)(P) &= \pi_1(\{s \mid \langle s, \pi_2\rangle \in P\} \cup \{\langle s, \rho\rangle \mid \langle s, \rho;\pi_2\rangle \in P\}), \\
(\pi_1 \Box \pi_2)(P) &= \pi_1(P) \cap \pi_2(P), \\
(\pi_1 \| \pi_2)(P) &= \pi_1(\{s \mid \langle s, \pi_2\rangle \in P\} \cup \{\langle s, \rho\rangle \mid \langle s, \rho\|\pi_2\rangle \in P\}) \cap \\
& \pi_2(\{s \mid \langle s, \pi_1\rangle \in P\} \cup \{\langle s, \rho\rangle \mid \langle s, \pi_1\|\rho\rangle \in P\}).
\end{aligned}
$$

Well-definedness of the box operator is immediate and is given below for reason of completeness.

5.10. Lemma. *The mapping '$\Box$' is well-defined and non-expansive.*

Proof. Choose $\pi_1, \pi_2 \in \Pi_2$ arbitrarily. Multiplicativity of $\pi_1 \Box \pi_2$ is straightforward. Suppose $\mathcal{D} \subseteq \mathcal{O}(\mathcal{S} + \mathcal{S} \times \Pi_2)$ is directed. Then we have

$$
\begin{aligned}
&(\pi_1 \Box \pi_2)(\bigcup \mathcal{D}) \\
&\quad = \bigcup\{\pi_1(P) \mid P \in \mathcal{D}\} \cap \bigcup\{\pi_2(P) \mid P \in \mathcal{D}\} && [\text{continuity } \pi_1, \pi_2] \\
&\quad = \bigcup\{\pi_1(P) \cap \pi_2(P) \mid P \in \mathcal{D}\} && [\mathcal{D} \text{ directed}] \\
&\quad = \bigcup\{(\pi_1 \Box \pi_2)(P) \mid P \in \mathcal{D}\}.
\end{aligned}
$$

Hence, $\pi_1 \Box \pi_2$ is open continuous and $\pi_1 \Box \pi_2 \in \Pi_2$. Non-expansiveness of $\Box$ is readily checked. $\Box$

Well-definedness of the operator ';' and '$\|$' is less trivial and can be supported by higher order transformations $\Omega_;$ and $\Omega_\|$, respectively.

5.11. Definition. Let $(\phi \in)\ Op = \Pi_2 \times \Pi_2 \xrightarrow{1} \Pi_2$. The higher-order transformation $\Omega_; : Op \to Op$ is given by

$$\Omega_;(\phi)(\pi_1, \pi_2)(P) \quad = \quad \pi_1(\{\theta \mid \phi'(\theta, \pi_2) \in P\})$$

where $\theta \in \Pi_2$, $\phi'(t, \pi_2) = \langle t, \pi_2 \rangle$, $\phi'(\langle t, \rho \rangle, \pi_2) = \langle t, \phi(\rho, \pi_2) \rangle$ and $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$.

We will prove that the operator ';' as defined in Definition 5.9 is the unique fixed point of $\Omega_;$. First we need three technical lemma's.

5.12. Lemma. *Let $\pi \in \Pi_2$. Put*

$$P' \quad = \quad \{s \in \mathcal{S} \mid \langle s, \pi \rangle \in P\}$$

for $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$. Then it holds that

(i) *If P is open then also P' is open.*
(ii) *If P is closed then also P' is closed.*
(iii) *For $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$, if $\bigcap \mathcal{X} \subseteq \bigcap \mathcal{Y}$ then $\bigcap\{P' \mid P \in \mathcal{X}\} \subseteq \bigcap\{P' \mid P \in \mathcal{Y}\}$.*

Proof. We only check part (i); parts (ii) and (iii) are straightforward. If $s \in P'$, then $\langle s, \pi \rangle \in P$. Hence, for some suitable $\epsilon < 1$, $B_\epsilon(\langle s, \pi \rangle) \subseteq P$ and therefore $B_\epsilon(s) \subseteq P'$, since $B_\epsilon(s) \subseteq \mathcal{S}$ (by the assumption $\epsilon < 1$). $\Box$

Notice that as direct consequence of Lemma 5.12.(iii) we have that if $P_1 \subseteq P_2$, then $P_1' \subseteq P_2'$ for all $P_1, P_2 \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$. Also, for $\mathcal{X} \subseteq \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$ we have

$$\bigcap\{P' \mid P \in \mathcal{X}\} \quad = \quad (\bigcap \mathcal{X})'.$$

5.13. Lemma. *Let $\phi : \Pi_2 \times \Pi_2 \to \Pi_2$ be non-expansive and $\pi \in \Pi$. Put*

$$P'' \quad = \quad \{\langle t, \rho \rangle \in \mathcal{S} \times \Pi_2 \mid \langle t, \phi(\rho, \pi) \rangle \in P\}$$

for $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)_1$. Then it holds that

(i) *If P is open then also P'' is open.*
(ii) *If P is closed then also P'' is closed.*
(iii) *For $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$, if $\bigcap \mathcal{X} \subseteq \bigcap \mathcal{Y}$ then $\bigcap\{P'' \mid P \in \mathcal{X}\} \subseteq \bigcap\{P'' \mid P \in \mathcal{Y}\}$.*

Proof. In verifying part (i) and (ii) we observe that P'' is the inverse image under the continuous map $\langle id, \lambda\rho.\, \phi(\rho, \pi) \rangle : (\mathcal{S} \times \Pi_2) \to (\mathcal{S} \times \Pi_2)$ of $P \cap (\mathcal{S} \times \Pi_2)$, which is open if P is open, and closed if P is closed. Part (iii) is readily checked. $\Box$

By Lemma 5.13.(iii) that if $P_1 \subseteq P_2$, then $P_1'' \subseteq P_2''$ for all $P_1, P_2 \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$. Also, for $\mathcal{X} \subseteq \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$ we have

$$\bigcap\{P'' \mid P \in \mathcal{X}\} \quad = \quad (\bigcap \mathcal{X})''.$$

5.14. Lemma. (i) *For any non-expansive $\phi : \Pi_2 \times \Pi_2 \to \Pi_2$, and $\pi_1, \pi_2 \in \Pi_2$ it holds that $\Omega_;(\phi)(\pi_1, \pi_2) \in \Pi_2$.*

(ii) *For any non expansive* $\phi : \Pi_2 \times \Pi_2 \to \Pi_2$ *the function* $\Omega_;$ *is also non-expansive. Moreover, for any* $\pi_1, \pi_2, \rho_1, \rho_2 \in \Pi_2$*:*

$$d(\Omega_;(\phi)(\pi_1,\pi_2), \Omega_;(\phi)(\rho_1,\rho_2)) \quad \leq \quad \max\{d(\pi_1,\rho_1), \tfrac{1}{2}d(\pi_2,\rho_2)\}.$$

(iii) *The transformation* $\Omega_;$ *is a* $\frac{1}{2}$*-contraction.*

Proof. (i) Let $\phi \in \Pi_2 \times \Pi_2 \to \Pi_2$ non-distance increasing, $\pi_1, \pi_2 \in \Pi_2$, and put $\pi = \Omega_;(\phi)(\pi_1,\pi_2)$. Let P', P'' be as in Lemmas 5.12 and 5.13 with respect to ϕ, π_2 and any P. We first verify $\Omega_;(\phi)(\pi_1,\pi_2) \in \Pi_2$.

(multiplicativity) We have for $\mathcal{X} \subseteq \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$

$$\bigcap\{P' \mid P \in \mathcal{X}\} = (\bigcap \mathcal{X})' \quad \text{and} \quad \bigcap\{P'' \mid P \in \mathcal{X}\} = (\bigcap \mathcal{X})''.$$

Hence, by disjointness of the P' and P'',

$$\bigcap\{P' \cup P'' \mid P \in \mathcal{X}\} \quad = \quad (\bigcap \mathcal{X})' \cup (\bigcap \mathcal{X})''.$$

Multiplicativity of π_1 now delivers

$$\bigcap\{\pi_1(P' \cup P'') \mid P \in \mathcal{X}\} \quad = \quad \pi_1((\bigcap \mathcal{X})' \cup (\bigcap \mathcal{X})'').$$

Since $\pi_1(P' \cup P'') = \pi(P)$ by definition, we have that $\Omega_;(\phi)(\pi_1,\pi_2)$ is multiplicative.

(continuity) Suppose $\mathcal{D} \subseteq \mathcal{O}(\mathcal{S} + \mathcal{S} \times \Pi_2)$ is directed. We then have

$$\begin{aligned}
&\pi(\bigcup \mathcal{D}) \\
&= \pi_1(\bigcup\{P' \mid P \in \mathcal{D}\} \cup \bigcup\{P'' \mid O \in \mathcal{D}\}) \\
&= \pi_1(\bigcup\{P' \cup P'' \mid P \in \mathcal{D}\}) \\
&= \bigcup\{\pi_1(P' \cup P'') \mid P \in \mathcal{D}\} && \text{[Lemmas 5.12, 5.13, continuity of } \pi_1] \\
&= \bigcup\{\pi(P) \mid P \in \mathcal{D}\}.
\end{aligned}$$

Hence, π is open-continuous.

(ii) Choose $\phi, \pi_1, \pi_2, \rho_1, \rho_2$ arbitrarily and put $\pi = \Omega_;(\phi)(\pi_1,\pi_2)$, and $\rho = \Omega_;(\phi)(\rho_1,\rho_2)$. We first establish

$$Q(s,\pi) = \{\phi'(\theta,\pi_2) \mid \theta \in Q(s,\pi_1)\} \tag{6}$$

for $s \in S$, using Lemma 4.4: Let $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$. Then

$$\begin{aligned}
&\{\phi'(\theta,\pi_2) \mid \theta \in Q(s,\pi_1)\} \subseteq P \\
&\Leftrightarrow \ \{\langle t,\pi_2\rangle \mid t \in Q(s,\pi_1)\} \cup \{\langle t, \phi(\pi',\pi_2)\rangle \mid \langle t,\pi'\rangle \in Q(s,\pi_1)\} \subseteq P \\
&\Leftrightarrow \ Q(s,\pi_1) \subseteq \{t \mid \langle t,\pi_2\rangle \in P\} \cup \{\langle t,\pi'\rangle \mid \langle t, \phi(\pi',\pi_2)\rangle \in P\} \\
&\Leftrightarrow \ Q(s,\pi_1) \subseteq P' \cup P'' && \text{[Definition of } P', P''] \\
&\Leftrightarrow \ s \in \pi_1(P' \cup P'') && \text{[Lemmas 4.4,5.12,5.13]} \\
&\Leftrightarrow \ s \in \pi(P). && \text{[Definition of } \pi]
\end{aligned}$$

A similar result holds for $Q(s,\rho)$. Now, let $s \in \mathcal{S}$. Then we have

$$\begin{aligned}
&d(Q(s,\pi), Q(s,\rho)) \\
&\leq \max\{d(\{\langle t,\pi_2\rangle \mid t \in Q(s,\pi_1)\}, \{\langle t,\rho_2\rangle \mid t \in Q(s,\rho_1)\}), \\
&\qquad d(\{\langle t,\phi(\pi',\pi_2)\rangle \mid \langle t,\pi'\rangle \in Q(s,\pi_1)\}, \{\langle t,\phi(\rho',\rho_2)\rangle \mid \langle t,\rho'\rangle \in Q(s,\rho_1)\})\} \\
&\leq \max\{d(Q(s,\pi_1), Q(s,\rho_1)), \tfrac{1}{2}d(\pi_2,\rho_2)\} \\
&\leq \max\{d(\pi_1,\rho_1), d(\pi_2,\rho_2)\}.
\end{aligned}$$

Taking the supremum over $\mathcal{S}$, the result follows.

(iii) Let $\phi_1, \phi_2 \in \Pi_2 \times \Pi_2 \to \Pi_2$ non-distance increasing. Then, for arbitrary $\pi_1, \pi_2 \in \Pi_2$, we have

$$\begin{aligned}
&d(Q(s,\Omega_;(\phi_1)(\pi_1,\pi_2)), Q(s,\Omega_;(\phi_2)(\pi_1,\pi_2))) \\
&\leq d(\{\langle t,\phi_1(\pi',\pi_2)\rangle \mid \langle t,\pi'\rangle \in Q(s,\pi_1)\}, \{\langle t,\phi_2(\pi',\pi_2)\rangle \mid \langle t,\pi'\rangle \in Q(s,\pi_1)\}) && \text{[Eq. 6]} \\
&\leq \tfrac{1}{2}d(\phi_1,\phi_2).
\end{aligned}$$

Hence $d(\Omega_;(\phi_1), \Omega_;(\phi_2)) \leq \frac{1}{2}d(\phi_1,\phi_2)$ and $\Omega_;$ is $\frac{1}{2}$-contractive. □

Now we are in a position to prove well-definedness of the operator ';': it is the unique fixed point of $\Omega_;$.

5.15. Theorem. *The operator ';' as defined in Definition 5.9 is the unique fixed point of* $\Omega_;$.

Proof. (Proof of Theorem 5.15.) Since $\Omega_;$ is a $\frac{1}{2}$-contraction it has a unique fixed point by Banach's fixed point theorem. It is easy to verify that ';' = $\mathit{fix}(\Omega_;)$. □

Next we proceed with the justification of the recursive definition of the operator '$\|$' in Π_2 given in Definition 5.9. We use again an higher-order transformation $\Omega_{\|} : Op \to Op$, where $Op = \Pi_2 \times \Pi_2 \xrightarrow{1} \Pi_2$.

5.16. Definition. Let $(\phi \in)\ Op = \Pi_2 \times \Pi_2 \xrightarrow{1} \Pi_2$. The higher-order transformation $\Omega_{\|} : Op \to Op$ is given by

$$\Omega_{\|}(\phi)(\pi_1, \pi_2) \quad = \quad \Omega_;(\phi)(\pi_1, \pi_2) \,\Box\, \Omega_;(\phi)(\pi_2, \pi_1)\,.$$

Next we see that the higher-order transformation $\Omega_{\|}$ is a contraction and has, assured by Banach's theorem, a unique fixed point, being —by definition— '$\|$'.

5.17. Theorem. (i) $\Omega_{\|}(\phi)$ *is non-expansive, for* $\phi \in Op$.
(ii) $\Omega_{\|}$ *is a* $\frac{1}{2}$*-contraction.*
Further, the operator '$\|$*' as defined in Definition 5.9 is the unique fixed point of* $\Omega_{\|}$.

Proof. (i) Straightforward from Lemma 5.10 and 5.14.(ii).
(ii) Direct from the definition of $\Omega_{\|}$ and Lemmas 5.10 and 5.14.(iii).
Since $\Omega_{\|}$ is a $\frac{1}{2}$-contraction it has a unique fixed point by Banach's fixed point theorem. It is easy to verify that '$\|$' = $\mathit{fix}(\Omega_{\|})$. □

Based on the $\mathcal{L}_0$-semantics wp_0 and the operators just given we are now ready to present the semantics wp_2 for $\mathcal{L}_2$.

5.18. Definition. The weakest precondition semantics wp_2 is the unique function in $\mathcal{L}_2 \to \Pi_2$ satisfying

$$\begin{array}{lcl}
wp_2[\![\langle d, A\rangle]\!] & = & \lambda P.\, wp_0[\![\langle \hat{d}, A\rangle]\!](P \cap \mathcal{S}),\\
wp_2[\![\langle d, x\rangle]\!] & = & wp_2[\![\langle d, d(x)\rangle]\!],\\
wp_2[\![\langle d, (S_1;S_2)\rangle]\!] & = & wp_2[\![\langle d, S_1\rangle]\!]; wp_2[\![\langle d, S_2\rangle]\!],\\
wp_2[\![\langle d, S_1 \Box S_2\rangle]\!] & = & wp_2[\![\langle d, S_1\rangle]\!] \Box wp_2[\![\langle d, S_2\rangle]\!],\\
wp_2[\![\langle d, S_1 \| S_2\rangle]\!] & = & wp_2[\![\langle d, S_1\rangle]\!] \| wp_2[\![\langle d, S_2\rangle]\!]
\end{array}$$

where $\hat{d}$ is some arbitrary declaration in $Decl_0$.

Some remarks are in order here. First, justification of the definition put forward can be obtained by another application of the higher order transformation technique. It is given in Lemma 5.19 below.

Second, the invocation of wp_0 enforces restriction of $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$ to $(P \cap \mathcal{S}) \in \mathcal{P}(\mathcal{S})$. Also the declaration $d \in Decl_2$, has formally to be adapted to a proper declaration $\hat{d} \in Decl_0$. However, since the atomic statements A do not contain any procedure variable, any declaration in $Decl_0$ will do.

The usage of the input/output semantics wp_0 for atomized statement A is intuitively clear. No interleaving possibilities are provided during execution of such atomized command (whence their name). This explains, for $\mathcal{L}_2$ (and $\mathcal{L}_3$), the contrast of the two sequential compositions $[A_1;A_2]$ and $(S_1;S_2)$:

$$\begin{array}{lcl}
wp_0[\![\langle d, [A_1;A_2]\rangle]\!] & : & \mathcal{P}(\mathcal{S}) \to \mathcal{P}(\mathcal{S}),\\
wp_2[\![\langle d, (S_1;S_2)\rangle]\!] & : & \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2) \to \mathcal{P}(\mathcal{S})
\end{array}$$

the former denying, the latter allowing interleaving with the environment.

The next lemma gives a justification of the semantics function wp_2 given in Definition 5.18. The construction makes essential use of the guardedness of procedure bodies. Let us at first extend the weight function $wgt_0 : \mathcal{L}_0 \to N$ to $wgt_2 : \mathcal{L}_2 \to N$ by adding to Definition 5.4 the clauses

$$\begin{array}{lcl} wgt_2(\langle d, [\mathcal{A}]\rangle) & = & 1, \\ wgt_2(\langle d, \mathcal{S}_1 \| \mathcal{S}_2\rangle) & = & \max\{wgt_2(\langle d, \mathcal{S}_1\rangle), wgt_2(\langle d, \mathcal{S}_2\rangle)\} + 1. \end{array}$$

5.19. Lemma. *Let $F \in Sem = \mathcal{L}_2 \to \Pi_2$. Define $\Psi_2 : Sem \to Sem$ inductively by*

$$\begin{array}{lcl} \Psi_2(F)(\langle d, [\mathcal{A}]\rangle)(P) & = & \lambda P.\, wp_0[\![\langle d', \mathcal{A}\rangle]\!](P \cap S) \text{ for } d' \in Decl_0, \\ \Psi_2(F)(\langle d, x\rangle) & = & \Psi_2(F)(\langle d, d(x)\rangle), \\ \Psi_2(F)(\langle d, S_1;S_2\rangle) & = & \Psi_2(F)(\langle d, S_1\rangle);F(\langle d, S_2\rangle), \\ \Psi_2(F)(\langle d, S_1 \Box S_2\rangle) & = & \Psi_2(F)(\langle d, S_1\rangle) \Box \Psi_2(F)(\langle d, S_2\rangle), \\ \Psi_2(F)(\langle d, S_1 \| S_2\rangle) & = & \Psi_2(F)(\langle d, S_1\rangle) \| \Psi_2(F)(\langle d, S_2\rangle). \end{array}$$

Then Ψ_2 is $\frac{1}{2}$-contractive and $wp_2[\![\cdot]\!]$ defined in Definition 5.18 is the unique fixed point of Ψ_2.

Proof. We show, by induction on $wgt_2(\langle d, S\rangle)$, that

$$d(\Psi_2(F_1)(\langle d, S\rangle), \Psi_2(F_2)(\langle d, S\rangle)) \quad \leq \quad \tfrac{1}{2} \cdot d(F_1, F_2)$$

for any $F_1, F_2 \in Sem$. We expand two typical sub-cases only.

$[x]$

$$\begin{array}{ll} & d(\Psi_2(F_1)(\langle d, x\rangle), \Psi_2(F_2)(\langle d, x\rangle) \\ = & d(\Psi_2(F_1)(\langle d, d(x)\rangle), \Psi_2(F_2)(\langle d, d(x)\rangle) \\ \leq & \frac{1}{2} d(F_1, F_2) \qquad \text{[induction hypothesis]} \end{array}$$

$[S_1;S_2]$

$$\begin{array}{ll} & d(\Psi_2(F_1)(\langle d, S_1;S_2\rangle), \Psi_2(F_2)(\langle d, S_1;S_2\rangle) \\ = & d(\Psi_2(F_1)(\langle d, S_1\rangle);F_1(\langle d, S_2\rangle), \Psi_2(F_2)(\langle d, S_1\rangle);F_2(\langle d, S_2\rangle)) \\ \leq & \max\{d(\Psi_2(F_1)(\langle d, S_1\rangle), \Psi_2(F_2)(\langle d, S_1\rangle)), \\ & \frac{1}{2} d(F_1(\langle d, S_2\rangle), F_2(\langle d, S_2\rangle))\} \qquad \text{[Lemma 5.14.(ii)]} \\ \leq & \frac{1}{2} d(F_1, F_2) \qquad \text{[induction hypothesis, definition } d(F_1, F_2)]. \end{array}$$

□

Let us, for the moment consider some examples, especially involving the parallel operator, to illustrate the semantics wp_2. The sequential statement $x := 1;y := 2$ will act as the first example. Maybe surprisingly we will obtain for the predicate

$$P \quad = \quad \{s \mid s(x) = 1 \,\&\, s(y) = 2\}$$

that $wp_2[\![x := 1;y := 2]\!](P) = \emptyset$ (for some fixed, but arbitrary and suppressed declaration d). Let us write π_1, π_2 for $wp_2[\![x := 1]\!]$, $wp_2[\![y := 2]\!]$, respectively. We then have

$$\begin{array}{ll} & wp_2[\![x := 1;y := 2]\!](P) \\ = & (\pi_1;\pi_2)(P) \\ = & \pi_1(\{s \mid \langle s, \pi_2\rangle \in P\} \cup \{\langle s, \rho\rangle \mid \langle s, \rho;\pi_2\rangle \in P\}) \\ = & \{s \mid \langle s\{1/x\}, \pi_2\rangle \in P\} \\ = & \emptyset. \end{array}$$

The point is that P only allows *immediate* terminating computations, where as the sequential composition has also one intermediate state as reflected in the pair $\langle s\{1/x\}, \pi_2\rangle$. Since, in general at the specification level, one will be indifferent as to how a final state will be reached, we can incorporate such pairs as $\langle s\{1/x\}, \pi_2\rangle$ in the predicate. So the predicate P has to be 'saturated' —enhanced with pairs representing the same input/output information— to accommodate for composite elements in $\mathcal{S} + \mathcal{S} \times \Pi_2$. We want the saturation P^* to satisfy

$$P^* \quad = \quad P \cup \{\langle s, \rho\rangle \mid s \in \rho(P^*)\}.$$

Hence P^* consists of elements of P and those pairs $\langle s, \rho\rangle$ for which s is appropriate in that it will lead to P^* following ρ. The predicate P^* can be defined as follows:

$$\begin{array}{rcl} P_0 & = & P \\ P_{k+1} & = & P_k \cup \{\langle s, \rho\rangle \mid s \in \rho(P_k)\} \\ P^* & = & \bigcup_k P_k. \end{array}$$

Turning back to the example for $x := 1;y := 2$ we now calculate the semantics for P^*.

$$\begin{array}{l} wp_2[\![x := 1;y := 2]\!](P^*) \\ = (\pi_1;\pi_2)(P^*) \\ = \pi_1(\{s \mid \langle s, \pi_2\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, \rho;\pi_2\rangle \in P^*\}) \\ = \{s \mid \langle s\{1/x\}, \pi_2\rangle \in P^*\} \\ = \{s \mid s\{1/x\} \in \pi_2(P^*)\} \\ = \{s \mid s\{1/x\}\{2/y\} \in P^*\} \\ = \{s \mid s\{1/x\}\{2/y\}(x) = 1 \;\&\; s\{1/x\}\{2/y\}(y) = 2\} \\ = \mathcal{S} \end{array}$$

which is the result to be expected.

As a second example, let us compute the semantics for $(x := 1\|y := 2)\|z := 3$ for $P = \{s \mid s(x) + s(y) = s(z)\}$ and P^* the saturation of P. Let, for shorthand, $\pi_1 = wp_2[\![x := 1]\!]$, $\pi_2 = wp_2[\![y := 2]\!]$ and $\pi_3 = wp_2[\![z := 3]\!]$. We have

$$\begin{array}{l} wp_2[\![(x := 1\|y := 2)\|z := 3]\!](P^*) \\ = ((\pi_1\|\pi_2)\|\pi_3)(P^*) \\ = (\pi_1\|\pi_2)(\{s \mid \langle s, \pi_3\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, \rho\|\pi_3\rangle \in P^*\})\cap \\ \quad \pi_3(\{s \mid \langle s, \pi_1\|\pi_2\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, (\pi_1\|\pi_2)\|\rho\rangle \in P^*\}) \\ = \pi_1(\{s \mid \langle s, \pi_2\rangle \in \{s \mid \langle s, \pi_3\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, \rho\|\pi_3\rangle \in P^*\}\}\cup \\ \qquad \{\langle s, \rho\rangle \mid \langle s, \rho\|\pi_2\rangle \in \{s \mid \langle s, \pi_3\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, \rho\|\pi_3\rangle \in P^*\}\})\cap \\ \quad \pi_2(\{s \mid \langle s, \pi_1\rangle \in \{s \mid \langle s, \pi_3\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, \rho\|\pi_3\rangle \in P^*\}\}\cup \\ \qquad \{\langle s, \rho\rangle \mid \langle s, \rho\|\pi_1\rangle \in \{s \mid \langle s, \pi_3\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, \rho\|\pi_3\rangle \in P^*\}\})\cap \\ \quad \pi_3(\{s \mid \langle s, \pi_1\|\pi_2\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, (\pi_1\|\pi_2)\|\rho\rangle \in P^*\}) \\ = \pi_1(\{s \mid \langle s, \pi_2\|\pi_3\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, (\rho\|\pi_2)\|\pi_3\rangle \in P^*\})\cap \\ \quad \pi_2(\{s \mid \langle s, \pi_1\|\pi_3\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, (\pi_1\|\rho)\|\pi_3\rangle \in P^*\})\cap \\ \quad \pi_3(\{s \mid \langle s, \pi_1\|\pi_2\rangle \in P^*\} \cup \{\langle s, \rho\rangle \mid \langle s, (\pi_1\|\pi_2)\|\rho\rangle \in P^*\}) \\ = \{s \mid s\{1/x\} \in (\pi_2\|\pi_3)(P^*)\}\cap\{s \mid s\{2/y\} \in (\pi_1\|\pi_3)(P^*)\}\cap\{s \mid s\{3/z\} \in (\pi_1\|\pi_2)(P^*)\} \\ = \ldots \\ = \{s \mid s\{1/x\}\{2/y\}\{3/z\} \in P^*\} \\ = \mathcal{S}. \end{array}$$

The example also indicates a more general relationship for parallel compositions of predicate transformers. Let us use $\pi_{1,..,k}$ to denote the left associated parallel composition of $\pi_1, \ldots, \pi_k$, $\pi^r_{1,..,k}$ (with $1 \leq r \leq k$) for $\pi_{1,..,k}$ but leaving out the operand π_r, and $\hat{\pi}^r_{1,..,k}$ for $\pi_{1,..,k}$ but now replacing the operand π_r by the predicate transformer ρ. We then have

$$\pi_{1,..,k}(P) \quad = \quad \bigcap_{r=1}^{k} \pi_r(\{s \mid \langle s, \pi^r_{1,..,k}\rangle \in P\} \cup \{\langle s, \rho\rangle \mid \langle s, \hat{\pi}^r_{1,..,k}\rangle \in P\}). \tag{7}$$

Equation (7) can be verified by a straightforward inductive argument using the various definitions.

As a more involving and likely more interesting application of the semantics wp_2, we treat the accumulator example (cf. [US93])

$$x := x + 2^0 \,\|\, x := x + 2^1 \,\|\, \ldots \,\|\, x := x + 2^n.$$

Under the assumption of atomic execution of the assignments $x := x + 2^i$, we want to calculate the weakest precondition for $P = \{s \mid s(x) = 2^{n+1}\}$. As discussed already, we first have to saturate the predicate yielding

$$P^* \quad = \quad \{s \mid s(x) = 2^{n+1}\} \cup \{\langle s, \rho\rangle \mid s \in \rho(P^*)\}.$$

Let, for convenience, $\pi_i = wp_2[\![x := x + 2^i]\!]$. So, we are heading, under the notational conventions given above, for $\pi_{0,..,n}(P^*)$. First we establish, for $0 \leq i_1 < \ldots < i_k \leq n$, the equation

$$\pi_{i_1,..,i_k}(P^*) \quad = \quad \{s \mid s(x) + \sum_{r=1}^{k} 2^{i_r} = 2^{n+1}\}, \tag{8}$$

by induction on k. We leave the base case, $k = 1$, to the reader. For the induction step, $k + 1$ we will employ equation (7).

$$\begin{array}{ll}
\pi_{i_1,..,i_{k+1}}(P^*) & \\
\quad = \bigcap_{r=1}^{k+1} \pi_{i_r}(\{s \mid \langle s, \pi^r_{i_1,..,i_k}\rangle \in P^*\} \cup \{\langle s,\rho\rangle \mid \langle s, \hat{\pi}^r_{i_1,..,i_k}\rangle \in P^*\}) & \text{[equality (7)]} \\
\quad = \bigcap_{r=1}^{k+1} \{s \mid \langle s\{\, s(x) + 2^{i_r}/x\,\}, \pi^r_{i_1,..,i_k}\rangle \in P^*\} & \text{[definition } \pi_{i_r}] \\
\quad = \bigcap_{r=1}^{k+1} \{s \mid s\{\, s(x) + 2^{i_r}/x\,\} \in \pi^r_{i_1,..,i_k}(P^*)\} & \text{[definition } P^*] \\
\quad = \bigcap_{r=1}^{k+1} \{s \mid (s(x) + 2^{i_r}) + (\sum_{q=1, q\neq r}^{k+1} 2^{i_q}) = 2^{n+1}\} & \text{[induction hypothesis]} \\
\quad = \bigcap_{r=1}^{k+1} \{s \mid s(x) + 2^{i_r}) + (\sum_{q=1}^{k+1} 2^{i_q}) = 2^{n+1}\} & \\
\quad = \{s \mid s(x) + 2^{i_r}) + (\sum_{q=1}^{k+1} 2^{i_q}) = 2^{n+1}\}. &
\end{array}$$

From equation (8) we immediate derive

$$\begin{array}{l}
\pi_{1,..,n}(P^*) \\
\quad = \{s \mid s(x) + (\sum_{i=1}^{n} 2^i) = 2^{n+1}\} \\
\quad = \{s \mid s(x) = (2^{n+1} - 1) = 2^{n+1}\} \\
\quad = \{s \mid s(x) = 1\}.
\end{array}$$

This concludes the examples for the semantics wp_2.

The semantics for the last language $\mathcal{L}_3$ will be obtained by combined application of the constructions used so far: meanings of atomic actions are delivered by the basic semantics wp_1 (as we referred to wp_0 for atomic commands in $\mathcal{L}_2$); recursion treated by body replacement and the underlying machinery of —presently unique— fixed points of environment transformations as developed for $\mathcal{L}_1$). First of all we have to define the semantic counterparts of '$;$', '$\Box_I$' and '$\|$' on the actual domain Π_3.

5.20. Definition. The operators '$;$', '$\Box_I$' and '$\|$' on Π_3 are given by

$$\begin{array}{lcl}
(\pi_1;\pi_2)(P) & = & \pi_1(\{s \mid \langle s, \pi_2\rangle \in cl(P)\} \cup \{\langle s,\rho\rangle \mid \langle s, \rho;\pi_2\rangle \in cl(P)\}), \\
(\Box_I \pi_i)(P) & = & \bigcap\{\pi_i(P) \mid i \in I\}, \\
(\pi_1 \| \pi_2)(P) & = & \pi_1(\{s \mid \langle s, \pi_2\rangle \in cl(P)\} \cup \{\langle s,\rho\rangle \mid \langle s, \rho\|\pi_2\rangle \in cl(P)\}) \cap \\
 & & \pi_2(\{s \mid \langle s, \pi_1\rangle \in P\} \cup \{\langle s,\rho\rangle \mid \langle s, \pi_1\|\rho\rangle \in P\}).
\end{array}$$

Note the striking similarity between Definition 5.20 and Definitions 5.9 and 5.6. Well definition of the above operators essentially follows the pattern as for Π_2. It is clear from the definition that $(\pi_1;\pi_2)(P) = (\pi_1;\pi_2)(cl(P))$ and that $(\pi_1\|\pi_2)(P) = (\pi_1\|\pi_2)(cl(P))$.

We are now well prepared for the definition of the semantics wp_3 for $\mathcal{L}_3$.

5.21. Definition. Let $(\eta \in)\, Env_3 = PVar \to \Pi_3$.

(i) The mapping $wp_3 : Stat_3 \to (Env_3 \to \Pi_3)$ is given by

$$\begin{array}{lcl} wp_3(A)(\eta) &=& \lambda P.\, wp_1[\![A]\!](P \cap \mathcal{S}), \\ wp_3(x)(\eta) &=& \eta(x), \\ wp_3(S_1;S_2)(\eta) &=& wp_3(S_1)(\eta);wp_3(S_2)(\eta), \\ wp_3(\Box_I\{S_i \mid i \in I\})(\eta) &=& \Box_I\{wp_3(S_i) \mid i \in I\}, \\ wp_3(S_1\|S_2)(\eta) &=& wp_3(S_1)(\eta)\| wp_3(S_2)(\eta). \end{array}$$

(ii) For all $d \in Decl_3$, the transformation $\mathrm{H}_d : Env_3 \to Env_3$ is given by

$$\mathrm{H}_d(\eta)(x) \quad = \quad wp_3(d(x))(\eta).$$

(iii) The weakest precondition semantics $wp_3 : \mathcal{L}_3 \to \Pi_3$ is given by

$$wp_3[\![\langle d, S\rangle]\!] \quad = \quad wp_3(S)(\eta_d)$$

where η_d is the unique fixed point of H_d.

Cumulatively, several remarks apply to the definition of the semantics wp_3 just given. First, for the treatment of atomic commands A using wp_3, we have to refrain form interleaving points (as we did for wp_2). In passing, since we treat declaration separately, both for $\mathcal{L}_3$ as for $\mathcal{L}_1$, there is no need here in choosing some (irrelevant) declaration for the restricted case. Well-definedness of wp_3 can be established by an appeal to the Banach's theorem on the contracting function H_d. We first need the following lemma.

5.22. Lemma. *For every $\langle d, S\rangle \in \mathcal{L}_3$ and $\eta_1, \eta_2 \in Env_3$ we have*

$$d_B(wp_3(\langle d, S\rangle)(\eta_1), wp_3(\langle d, S\rangle)(\eta_2)) \quad \leq \quad d(\eta_1, \eta_2)$$

Proof. The proof goes by structural induction on the syntactic complexity of the statements. We expand two typical sub-cases only.

$[x]$

$$\begin{array}{l} d_B(wp_3(\langle d, x\rangle)(\eta_1), wp_3(\langle d, x\rangle)(\eta_2)) \\ \quad = d_B(\eta_1(x)), \eta_2(x)) \\ \quad \leq d(\eta_1, \eta_2) \end{array}$$

$[S_1;S_2]$

$$\begin{array}{ll} d_B(wp_3(\langle d, S_1;S_2\rangle)(\eta_1), wp_3(\langle d, S_1;S_2\rangle)(\eta_2)) & \\ \quad = d_B(wp_3(\langle d, S_1\rangle)(\eta_1);wp_3(\langle d, S_2\rangle)(\eta_1), wp_3(\langle d, S_1\rangle)(\eta_2);wp_3(\langle d, S_2\rangle)(\eta_2)) & \\ \quad \leq \max\{d_B(wp_3(\langle d, S_1\rangle)(\eta_1), wp_3(\langle d, S_1\rangle)(\eta_2)), & \\ \qquad \frac{1}{2}\cdot d_B(wp_3(\langle d, S_2\rangle)(\eta_1), wp_3(\langle d, S_2\rangle)(\eta_2))\} & \text{[Lemma 5.14.(ii)]} \\ \quad \leq \max\{d(\eta_1, \eta_2), \frac{1}{2}\cdot d(\eta_1, \eta_2)\} & \text{[induction hypothesis]} \\ \quad = d(\eta_1, \eta_2). & \Box \end{array}$$

Next we show that $\mathrm{H}_d : Env_3 \to Env_3$ is a contraction and hence — by Banach's theorem and completeness of Env_3 — it has unique fixed point, say η_d. Notice that η_d is then the unique environment such that $wp_3(\langle d, x\rangle)(\eta) = wp_3(\langle d, d(x)\rangle)(\eta)$.

5.23. Lemma. *For every $d \in Decl$ the function $\mathrm{H}_d : Env_3 \to Env_3$ is $\frac{1}{2}$-contracting, that is*

$$d(\mathrm{H}_d(\eta_1), \mathrm{H}_d(\eta_2)) \quad \leq \quad \tfrac{1}{2}\cdot d(\eta_1, \eta_2)$$

for every $\eta_1, \eta_2 \in Env_3$.

Proof. Since $\mathrm{H}_d(\eta)(x) = wp_3(\langle d, d(x)\rangle)(\eta)$ and $d(x) \in GStat_3$ it is enough to prove for every $x \in PVar$ and guarded statement $G \in GStat_3$ that

$$d_B(wp_3(\langle d, G\rangle)(\eta_1), wp_3(\langle d, G\rangle)(\eta_2)) \quad \leq \quad \tfrac{1}{2} \cdot d(\eta_1, \eta_2)$$

The proof proceed by structural induction on the syntactic complexity of guarded statements. We treat one sub-case only

$$\begin{array}{lll}
[G_1;S_2] & d_B(wp_3(\langle d, G_1;S_2\rangle)(\eta_1), wp_3(\langle d, G_1;S_2\rangle)(\eta_2)) & \\
& = d_B(wp_3(\langle d, G_1\rangle)(\eta_1);wp_3(\langle d, S_2\rangle)(\eta_1), wp_3(\langle d, G_1\rangle)(\eta_2);wp_3(\langle d, S_2\rangle)(\eta_2)) & \\
& \leq \max\{d_B(wp_3(\langle d, G_1\rangle)(\eta_1), wp_3(\langle d, G_1\rangle)(\eta_2)), & \\
& \quad \tfrac{1}{2} \cdot d_B(wp_3(\langle d, S_2\rangle)(\eta_1), wp_3(\langle d, S_2\rangle)(\eta_2))\} & \text{[Lemma 5.14.(ii)]} \\
& \leq \max\{\tfrac{1}{2} \cdot d(\eta_1, \eta_2), \tfrac{1}{2} \cdot d(\eta_1, \eta_2)\} & \text{[induction hypothesis, Lemma 5.22]} \\
& = \tfrac{1}{2} \cdot d(\eta_1, \eta_2). & \square
\end{array}$$

Finally we have the following aspected result on $wp_3[\![\cdot]\!]\mathcal{L}_3 \to \Pi_3$.

5.24. Lemma. *The predicate transformer semantics $wp_3[\![\cdot]\!]$ as defined in Definition 5.21 is the unique function in $\mathcal{L}_3 \to \Pi_3$, satisfying*

$$\begin{array}{lll}
wp_3[\![\langle d, [\mathcal{A}]\rangle]\!] & = & \lambda P.\, wp_1[\![\langle d', \mathcal{A}\rangle]\!](P \cap S) \textit{ for } d' \in Decl_1, \\
wp_3[\![\langle d, x\rangle]\!] & = & wp_3[\![\langle d, d(x)\rangle]\!], \\
wp_3[\![\langle d, S_1;S_2\rangle]\!] & = & wp_3[\![\langle d, S_1\rangle]\!];wp_3[\![\langle d, S_2\rangle]\!], \\
wp_3[\![\langle d, \Box_I\{S_i \mid i \in I\}\rangle]\!] & = & \Box_I\{wp_3[\![\langle d, S_i\rangle]\!] \mid i \in I\}, \\
wp_3[\![\langle d, S_1 \| S_2\rangle]\!] & = & wp_3[\![\langle d, S_1\rangle]\!] \| wp_3[\![\langle d, S_2\rangle]\!].
\end{array}$$

Proof. The proof can be adapted from that of Lemma 5.8 and hence is omitted. $\square$

6 Refinement, Simulation and State Transformers

The refinement relation given in Definition 4.15 is closely related to the notion of simulation for transition systems in a sense that we will explain below. The relationship will be based on four isomorphisms, to be given in this section, between the predicate transformer domains Π_i, and the state transformers domains Σ_i. The domains Σ_0 and Σ_1 are based on the Smyth power domain, while the domains Σ_2 and Σ_3 are (variations of) the metric resumption domains of De Bakker and Zucker [BZ82].

We start with the definition of (two versions of) the Smyth power domain.

6.1. Definition. Let $\mathcal{S}$ be a set of state and let $\mathcal{S}_\perp$ be the set $\mathcal{S} \cup \{\perp\}$ ordered by $x \leq y$ if and only if $x =\perp$ or $x = y$. The Smyth power domain of $\mathcal{S}_\perp$ is defined as the set

$$\mathcal{SP}(\mathcal{S}_\perp) = \{A \mid A \subseteq \mathcal{S}\} \cup \{\mathcal{S}_\perp\}$$

ordered by the superset order. The *finitary* Smyth power domain of $\mathcal{S}_\perp$ is defined as the set

$$\mathcal{SP}_{co}(S_\perp) = \{A \mid A \subseteq_{fin} \mathcal{S}\} \cup \{\mathcal{S}_\perp\}$$

ordered by the superset order.

The state transformers domains Σ_0 and Σ_1 are given by $\Sigma_0 = \mathcal{S} \to \mathcal{SP}_{co}(\mathcal{S}_\perp)$ and $\Sigma_1 = \mathcal{S} \to \mathcal{SP}(\mathcal{S}_\perp)$. They are ordered by pointwise extension of the superset ordering on the Smyth power domain. Now, statements are viewed as functions. Values in $\mathcal{P}(\mathcal{S})$ represent the results of terminating non-deterministic computations, while the divergent ones are mapped to $\{\mathcal{S}_\perp\}$. The empty set is interpreted as deadlock.

On Σ_0 and Σ_1 we have the usual operations ';' and '$\Box_I$'.

6.2. Definition. Let I be a non-empty index set I. Define the operators ';' and '$\Box_I$' on Σ_1 by

$$
\begin{array}{lcl}
(\sigma_1;\sigma_2)(s) & = & \left\{ \begin{array}{ll} \bigcup\{\sigma_2(t) \mid t \in \sigma_1(s)\} & \text{if } \sigma_1(s) \neq \mathcal{S}_\perp \text{ and } \forall t \in \sigma_1(s).\ \sigma_2(t) \neq \mathcal{S}_\perp \\ \mathcal{S}_\perp & \text{otherwise.} \end{array} \right. \\
(\Box_I\{\sigma_i \mid i \in I\})(s) & = & \bigcup\{\sigma_i(s) \mid i \in I.\}
\end{array}
$$

The relationships between predicate transformers in Π_0 or Π_1 and state transformers in Σ_0 or Σ_1 is well-known ([Plo79, Bac81, BK94b]) and is repeated in the theorem below for reason of completeness.

6.3. Theorem. *The function $pt : \Sigma_1 \to \Pi_1$ defined by*

$$pt(\sigma)(P) = \{s \in \mathcal{S} \mid \sigma(s) \subseteq P\}$$

forms an order-isomorphism between Σ_1 and Π_1 with inverse inverse pt^{-1} given by

$$pt^{-1}(\pi)(s) \quad = \quad \left\{ \begin{array}{ll} \bigcap\{P \mid s \in \pi(P)\} & \textit{if } s \in \pi(\mathcal{S}) \\ \Sigma_\perp & \textit{otherwise} \end{array} \right.$$

Also, it restrict to an order isomorphism between Σ_0 and Π_0. Moreover, for every non-empty index set I we have
(i) $pt(\sigma_1;\sigma_2) = pt(\sigma_1);pt(\sigma_2)$,
(ii) $pt(\Box_I\{\sigma_i \mid i \in I\}) = \Box_I\{pt(\sigma_i) \mid i \in I\}$.
where σ_1, σ_2 and $\sigma_i \in \Sigma_1$ for all $i \in I$. □

A refinement relation preserving partial and total correctness is defined in [Bac80] for the domains Π_0 and Π_1. It coincides with pointwise extension of the subset order in $\mathcal{P}(\mathcal{S})$. Since the isomorphism $pt : \Sigma_1 \to \Pi_1$ is an order isomorphism, the above theorem gives also a state transformer representation of the Back's refinement relation: the pointwise extension of the order of the Smyth power domain.

Next we restrict our attention to the language $\mathcal{L}_2$ of Definition 3.3. Let us define the syntactic refinement relation '$\trianglelefteq$' by

$$
\begin{array}{rll}
S_1 \trianglelefteq S_2 & \text{if} & wp_2[\![\langle d, S_1\rangle]\!] \trianglelefteq wp_2[\![\langle d, S_2\rangle]\!] \\
S_1 \mathrel{\trianglelefteq\!\!\trianglerighteq} S_2 & \text{if} & wp_2[\![\langle d, S_1\rangle]\!] \trianglelefteq wp_2[\![\langle d, S_2\rangle]\!]\ \&\ wp_2[\![\langle d, S_2\rangle]\!] \trianglelefteq wp_2[\![\langle d, S_1\rangle]\!]
\end{array}
$$

for a fixed declaration $d \in Decl_2$ and $S_1, S_2 \in Stat_2$. By Definitions 5.18 and 4.15 we have immediately that $A_1 \trianglelefteq A_2$ if and only if $wp_0[\![\langle d, A_1\rangle]\!](P) \subseteq wp_0[\![\langle d, A_2\rangle]\!](P)$ for all $P \subseteq \mathcal{S}$ and $d \in Decl_0$. In other words, $A_1 \trianglelefteq A_2$ if and only if A_1 is refined by A_2 according the Back's refinement relation. In Figure 2 some syntactic refinement rules are given for atomic statements in $\mathcal{L}_2$, while in Figure 1 some basic refinement rules for general statements in $\mathcal{L}_2$ are given. Their soundness can be proved by a simple application of Definitions 4.15 and 5.18.

Turning back to the isomorphisms between state and predicate transformers, we define Σ_2 as a variation of the domain introduced in [BZ82]. Let $\mathcal{S}$ be a set of states and consider the domain Σ_2 given as unique (up to isometries) solution of the domain equation

$$X \quad = \quad \mathcal{S} \to (\mathcal{P}_{co}(\mathcal{S} + \mathcal{S} \times \tfrac{1}{2}(X))).$$

(Id)	$S \trianglelefteq S$
$(Magic)$	$S \trianglelefteq$ **false**$\rightarrow$
$(Trans)$	$\dfrac{S_1 \trianglelefteq S_2\ \&\ S_2 \trianglelefteq S_3}{S_1 \trianglelefteq S_3}$

Figure 1: Some general refinement rules for $\mathcal{L}_2$

$(LeftAtSkip)$	$[\mathbf{true}\!\rightarrow ; A] \trianglelefteq\!\trianglerighteq A$
$(RightAtSkip)$	$[A;\mathbf{true}\!\rightarrow] \trianglelefteq\!\trianglerighteq A$
$(AssAtSeq)$	$[A_1;[A_2;A_3] \trianglelefteq\!\trianglerighteq [[A_1;A_2];A_3]$
$(AtBox)$	$[A_1 \Box A_2] \trianglelefteq A_1$
$(IdemAtBox)$	$A \trianglelefteq [A \Box A]$
$(CommAtBox)$	$[A_1 \Box A_2] \trianglelefteq [A_2 \Box A_1]$
$(AssAtBox)$	$[A_1 \Box [A_2 \Box A_3]] \trianglelefteq\!\trianglerighteq [[A_1 \Box A_2] \Box A_3]$
$(Test)$	$\dfrac{\forall s \in \mathcal{S}.\ \mathcal{B}(b_1)(s) \Rightarrow \mathcal{B}(b_2)(s)}{b_2\!\rightarrow\ \trianglelefteq b_1\!\rightarrow}$
$(CompAtSeq)$	$\dfrac{A_1 \trianglelefteq A_1' \ \&\ A_2 \trianglelefteq A_2'}{[A_1;A_2] \trianglelefteq [A_1';A_2']}$
$(CompAtBox)$	$\dfrac{A_1 \trianglelefteq A_1' \ \&\ A_2 \trianglelefteq A_2'}{[A_1 \Box A_2] \trianglelefteq [A_1' \Box A_2']}$

Figure 2: Some refinement rules for atomic statements in $\mathcal{L}_2$

The intuition behind the above domain is as follows: statements are functions which deliver for each input state a set consisting of output states and/or pairs composed by the output after one atomic computation step together with a function representing the rest of the computation. The domain Σ_2 comes equipped with the following operations (cf. [BZ82]).

6.4. Definition. The operators ';', '$\Box$' and '$\|$' on Σ_2 are given by

$$\begin{array}{rcl} \sigma_1;\sigma_2(s) & = & \{\langle t, \sigma_2\rangle \mid t \in \sigma_1(s)\} \cup \{\langle t, \tau;\sigma_2\rangle \mid \langle t, \tau\rangle \in \sigma_1(s)\}, \\ \sigma_1 \Box \sigma_2(s) & = & \sigma_1(s) \cup \sigma_2(s), \\ \sigma_1 \| \sigma_2(s) & = & \{\langle t, \sigma_2\rangle \mid t \in \sigma_1(s)\} \cup \{\langle t, \tau \| \sigma_2\rangle \mid \langle t, \tau\rangle \in \sigma_1(s)\} \cup \\ & & \{\langle t, \sigma_1\rangle \mid t \in \sigma_2(s)\} \cup \{\langle t, \sigma_1 \| \tau\rangle \mid \langle t, \tau\rangle \in \sigma_2(s)\}, \end{array}$$

for $\sigma_1, \sigma_2 \in \Sigma_2$ and $s \in \mathcal{S}$.

On the basis of the general isomorphisms between state an predicate transformers studied in [BK94a] we can formulate the relationship between Σ_2 and Π_2. The two domains are isomorphic. However, the isomorphism is not an order-isomorphism as for pt from the complete partial order Σ_1 to the complete partial order Π_1, but an isometry from the complete metric space Σ_2 to the complete metric space Π_2. We need some steps preparatory to prove the isomorphism.

6.5. Definition. Define the domain transformation $pt : \Sigma_2 \rightarrow \Pi_2$ by

$$pt(\sigma)(P) \quad = \quad \{s \in \mathcal{S} \mid (t \in \sigma(s) \Rightarrow t \in P)\ \&\ (\langle t, \tau\rangle \in \sigma(s) \Rightarrow \langle t, pt(\tau)\rangle \in P)\},$$

for all $\sigma \in \Sigma_2$, and $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$.

In order to justify the well-definition of pt we introduce an higher-order transformation in a such way that it is contractive and pt is its unique fixed point. Define the higher-order transformation $\Omega_{pt} : (\Sigma_2 \xrightarrow{1} \Pi_2) \rightarrow (\Sigma_2 \xrightarrow{1} \Pi_2)$ by

$$\Omega_{pt}(tr)(\sigma)(P) \quad = \quad \{s \in \mathcal{S} \mid (t \in \sigma(s) \Rightarrow t \in P)\ \&\ (\langle t, \tau\rangle \in \sigma(s) \Rightarrow \langle t, tr(\tau)\rangle \in P)\},$$

for all $tr : \Sigma_2 \xrightarrow{1} \Pi_2$, $\sigma \in \Sigma_2$, and $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$.

6.6. Lemma. *Let* $tr : \Sigma_2 \xrightarrow{1} \Pi_2$ *and* $\sigma \in \Sigma_2$*. Then*

(i) $\Omega_{pt}(tr)(\sigma) \in \Pi_2$,

(ii) $\Omega_{pt}(tr)$ *is non-expansive,*

(iii) *the transformation* Ω_{pt} *is* $\frac{1}{2}$*-contractive.*

Therefore Ω_{pt} *has a unique fixed point which is the function* pt *of Definition 6.5.*

Proof. (i) Choose arbitrary $tr : \Sigma_2 \xrightarrow{1} \Pi_2$ and $\sigma \in \Sigma_2$, and put $\pi = \Omega_{pt}(tr)(\sigma)$. Multiplicativity of π is straightforwardly checked. Continuity of π is verified, with $\mathcal{D} \subseteq \mathcal{O}(\mathcal{S} + \mathcal{S} \times \Pi_2)$ directed, as follows. Note, that the set

$$\{t \in \mathcal{S} \mid t \in \sigma(s)\} \quad \cup \quad \{\langle t, tr(\tau)\rangle \in \mathcal{S} \times \Pi_2 \mid \langle t, \tau\rangle \in \sigma(s)\}$$

is compact: its first constituent equals $\sigma(s) \cap \mathcal{S}$, which is the intersection of a compact and a closed set. Its second constituent is the continuous image – under (id, tr) – of the compact set $\sigma(s) \cap (\mathcal{S} \times \Sigma_2)$. Therefore,

$$\begin{aligned}
& s \in \pi(\textstyle\bigcup \mathcal{D}) \\
\Leftrightarrow\ & \{t \in \mathcal{S} \mid t \in \sigma(s)\} \cup \{\langle t, tr(\tau)\rangle \in \mathcal{S} \times \Pi_2 \mid \langle t, \tau\rangle \in \sigma(s)\} \subseteq \textstyle\bigcup \mathcal{D} && [\text{definition } \Omega_{pt}] \\
\Leftrightarrow\ & \exists P \in \mathcal{D}.\ \{t \mid t \in \sigma(s)\} \cup \{\langle t, tr(\tau)\rangle \mid \langle t, \tau\rangle \in \sigma(s)\} \subseteq P && [\text{compactness}] \\
\Leftrightarrow\ & s \in \textstyle\bigcup \{\pi(P) \mid P \in \mathcal{D}\} \ .
\end{aligned}$$

(ii) For arbitrary tr, σ, we first show

$$Q(s, \Omega_{pt}(tr)(\sigma)) = \{t \in \mathcal{S} \mid t \in \sigma(s)\} \cup \{\langle t, tr(\tau)\rangle \in \mathcal{S} \times \Pi_2 \mid \langle t, \tau\rangle \in \sigma(s)\}\,. \tag{9}$$

Using Q' as shorthand for the right-hand side of the equality we have, for any $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$:

$$\begin{aligned}
& s \in \Omega_{pt}(tr)(\sigma)(P) \\
\Leftrightarrow\ & (t \in \sigma(s) \Rightarrow t \in P)\ \&\ (\langle t, \tau\rangle \in \sigma(s) \Rightarrow \langle t, tr(\tau)\rangle \in P) && [\text{definition } \Omega_{pt}] \\
\Leftrightarrow\ & Q' \subseteq P \ .
\end{aligned}$$

Therefore, by Lemma 4.4 the claim 9 follows. Now, let $tr : \Sigma_2 \xrightarrow{1} \Pi_2$ and choose, in order to show the non-expansiveness of $\Omega_{pt}(tr)$, $\sigma_1, \sigma_2 \in \Sigma_2$. We then have

$$\begin{aligned}
& d(\Omega_{pt}(tr)(\sigma_1), \Omega_{pt}(tr)(\sigma_2)) \\
=\ & \sup\{d(Q(s, \Omega_{pt}(tr)(\sigma_1)), Q(s, \Omega_{pt}(tr)(\sigma_2))) \mid s \in \mathcal{S}\} \\
=\ & \max\{\sup\{d(\{t \in \mathcal{S} \mid t \in \sigma_1(s)\}, \{t \in \mathcal{S} \mid t \in \sigma_2(s)\}) \mid s \in \mathcal{S}\}, \\
& \sup\{d(\{\langle t, tr(\tau)\rangle \mid \langle t, \tau\rangle \in \sigma_1(s)\}, \{\langle t, tr(\tau)\rangle \mid \langle t, \tau\rangle \in \sigma_2(s)\} \mid s \in \mathcal{S}\}\} && [\text{Equation 9}] \\
\leq\ & \sup\{d(\sigma_1(s), \sigma_2(s) \mid s \in \mathcal{S}\} && [tr \text{ non-expansive}] \\
=\ & d(\sigma_1, \sigma_2) \ .
\end{aligned}$$

(iii) Pick any $tr_1, tr_2 \in \Sigma_2 \xrightarrow{1} \Pi_2$. Then we have

$$\begin{aligned}
& d(\Omega_{pt}(tr_1), \Omega_{pt}(tr_2)) \\
=\ & \sup\{d(\Omega_{pt}(tr_1)(\sigma), \Omega_{pt}(tr_2)(\sigma)) \mid \sigma \in \Sigma_2\} && [\text{definition of } d \text{ on } \Sigma_2 \to \Pi_2] \\
=\ & \sup\{d(Q(s, \Omega_{pt}(tr_1)(\sigma)), Q(s, \Omega_{pt}(tr_2)(\sigma))) \mid \sigma \in \Sigma_2\ \&\ s \in \mathcal{S}\} && [\text{definition of } d \text{ on } \Pi_2] \\
\leq\ & \sup\{d(\{\langle t, tr_1(\tau)\rangle \mid \langle t, \tau\rangle \in \sigma(s)\}, \\
& \{\langle t, tr_2(\tau)\rangle \mid \langle t, \tau\rangle \in \sigma(s)\}) \mid \sigma \in \Sigma_2\ \&\ s \in \mathcal{S}\} && [\text{Equation 9}] \\
\leq\ & \sup\{\tfrac{1}{2} d(tr_1(\tau), tr_2(\tau)) \mid \sigma \in \Sigma_2\ \&\ \langle t, \tau\rangle \in \sigma(s)\} \\
\leq\ & \tfrac{1}{2} d(tr_1, tr_2) \ .
\end{aligned}$$

□

6.7. Corollary. *For* $\sigma \in \Sigma_2$ *and* $s \in \mathcal{S}$*, it holds that*

$$Q(s, pt(\sigma)) \quad = \quad \{t \in \mathcal{S} \mid t \in \sigma(s)\} \cup \{\langle t, pt(\tau)\rangle \in \mathcal{S} \times \Pi_2 \mid \langle t, \tau\rangle \in \sigma(s)\}\,.$$

Proof. From the proof of the Lemma 6.6 we have, taking pt for tr,

$$Q(s, \Omega_{pt}(pt)(\sigma)) \quad = \quad \{t \in \mathcal{S} \mid t \in \sigma(s)\} \cup \{\langle t, pt(\tau)\rangle \in \mathcal{S} \times \Pi_2 \mid \langle t, \tau\rangle \in \sigma(s)\}$$

from which the conclusion follows since pt is the unique fixed point of Ω_{pt}. □

Next we define a function which transforms a predicate transformer in a state transformer.

6.8. Definition. Define the domain transformation $st : \Pi_2 \to \Sigma_2$ by

$$st(\pi)(s) \quad = \quad \{t \in \mathcal{S} \mid t \in Q(s,\pi)\} \cup \{\langle t, st(\rho)\rangle \in \mathcal{S} \times \Sigma_2 \mid \langle t,\rho\rangle \in Q(s,\pi)\},$$

for all $\pi \in \Pi_2$, and $s \in \mathcal{S}$.

Again, to show the well-definition of the function st we introduce the higher-order transformation $\Omega_{st} : (\Pi_2 \xrightarrow{1} \Sigma_2) \to (\Pi_2 \xrightarrow{1} \Sigma_2)$. It is given by

$$\Omega_{st}(tr)(\pi)(s) \quad = \quad \{t \in \mathcal{S} \mid t \in Q(s,\pi)\} \cup \{\langle t, tr(\rho)\rangle \in \mathcal{S} \times \Sigma_2 \mid \langle t,\rho\rangle \in Q(s,\pi)\},$$

for all $tr : \Pi_2 \xrightarrow{1} \Sigma_2$, $\pi \in \Pi_2$, and $s \in \mathcal{S}$.

6.9. Lemma. *Let $tr : \Pi_2 \xrightarrow{1} \Sigma_2$ and $\pi \in \Pi_2$. Then*
(i) $\Omega_{st}(tr)(\pi) \in \Sigma_2$,
(ii) $\Omega_{st}(tr)$ *is non-expansive, and*
(iii) *the transformation Ω_{st} is $\frac{1}{2}$-contractive.*

Proof. Similar to the proof of Lemma 6.6 and omitted. □

We are now in a position to prove the isometry between Σ_2 and Π_2.

6.10. Theorem. *The functions $pt : \Sigma_2 \xrightarrow{1} \Pi_2$ and $st : \Pi_2 \to \Sigma_2$ form an isometry between Σ_2 and Π_2.*

Proof. It is enough to check that $pt : \Sigma_2 \to \Pi_2$ and $st : \Pi_2 \to \Sigma_2$ satisfy

$$st \circ pt = id_{\Sigma_2} \quad \text{and} \quad pt \circ st = id_{\Pi_2} .$$

First we verify $st \circ pt = id_{\Sigma_2}$. Note that by the respective definitions and Corollary 6.7,

$$\begin{aligned}
&st(pt(\sigma))(s) \\
&\quad = \{t \in \mathcal{S} \mid t \in Q(s,pt(\sigma))\} \cup \{\langle t, st(\rho)\rangle \in \mathcal{S} \times \Sigma_2 \mid \langle t,\rho\rangle \in Q(s,pt(\sigma))\} \\
&\quad = \{t \in \mathcal{S} \mid t \in \sigma(s)\} \cup \{\langle t, st(pt(\tau))\rangle \in \mathcal{S} \times \Sigma_2 \mid \langle t,\tau\rangle \in \sigma(s)\} \ .
\end{aligned}$$

Suppressing the subscript Σ_2 for the moment, by the above characterization of $st(pt(\sigma))(s)$ we derive

$$\begin{aligned}
&d(id, st \circ pt) \\
&\quad = \sup\{d(\sigma(s), st(pt(\sigma))(s)) \mid \sigma \in \Sigma_2 \ \&\ s \in \mathcal{S}\} \\
&\quad \leq \sup\{d(\langle t,\tau\rangle, \langle t, st(pt(\tau))\rangle) \mid \sigma \in \Sigma_2 \ \&\ s \in \mathcal{S} \ \&\ \langle t,\tau\rangle \in \sigma(s)\} \\
&\quad \leq \tfrac{1}{2}\sup\{d(\tau, st(pt(\tau))) \mid \sigma \in \Sigma_2 \ \&\ s \in \mathcal{S} \ \&\ \langle t,\tau\rangle \in \sigma(s)\} \\
&\quad \leq \tfrac{1}{2} d(id, st \circ pt) \ .
\end{aligned}$$

From this we conclude $st \circ pt = id_{\Sigma_2}$. Likewise, but slightly simpler, one derives

$$d(id_{\Pi_2}, pt \circ st) \quad \leq \quad \tfrac{1}{2} d(id_{\Pi_2}, pt \circ st)$$

and, consequently, $pt \circ st = id_{\Pi_2}$. □

Next we prove that both pt and st preserve the operations sequential composition, choice and merge. Since they form an isomorphism, it is enough to prove the result only for pt.

6.11. Lemma. *Let $\sigma_1, \sigma_2 \in \Sigma_2$. Then*
(i) $pt(\sigma_1;\sigma_2) = pt(\sigma_1);pt(\sigma_2)$;

$(AssSeq)$	$S_1;(S_2;S_3) \trianglelefteq\!\trianglerighteq (S_1;S_2);S_3$	
(Box)	$S_1 \Box S_2 \trianglelefteq S_1$	
$(IdemBox)$	$S \trianglelefteq S \Box S$	
$(CommBox)$	$S_1 \Box S_2 \trianglelefteq S_2 \Box S_1$	
$(AssBox)$	$S_1 \Box (S_2 \Box S_3) \trianglelefteq\!\trianglerighteq (S_1 \Box S_2) \Box S_3$	
$(CommPar)$	$S_1 \| S_2 \trianglelefteq S_2 \| S_1$	
$(AssPar)$	$S_1 \| (S_2 \| S_3) \trianglelefteq\!\trianglerighteq (S_1 \| S_2) \| S_3$	
$(PVar1)$	$\dfrac{d(x) \trianglelefteq S}{x \trianglelefteq S}$	
$(PVar2)$	$\dfrac{d(x) \trianglelefteq d(y)^x_y}{x \trianglelefteq y}$	where S^x_y is S with x replacing y.

Figure 3: Some refinement rules for statements in $\mathcal{L}_2$

(ii) $pt(\sigma_1 \Box \sigma_2) = pt(\sigma_1) \Box pt(\sigma_2)$;

(iii) $pt(\sigma_1 \| \sigma_2) = pt(\sigma_1) \| pt(\sigma_2)$.

Proof. (i) First notice the following

$$\begin{array}{ll} s \in \pi_1;\pi_2(P) & \\ \quad \Leftrightarrow \; s \in \pi_1(\{t \in \mathcal{S} \mid \langle t, \pi_2\rangle \in P\} \cup \{\langle t, \rho\rangle \mid \langle t, \rho;\pi_2\rangle \in P\}) & \\ \quad \Leftrightarrow \; Q(s, \pi_1) \subseteq \{t \in \mathcal{S} \mid \langle t, \pi_2\rangle \in P\} \cup \{\langle t, \rho\rangle \mid \langle t, \rho;\pi_2\rangle \in P\} & [\text{Lemma 4.4}] \\ \quad \Leftrightarrow \; \{\langle t, \pi_2\rangle \mid t \in Q(s, \pi_1)\} \cup \{\langle t, \rho;\pi_2\rangle \mid \langle t, \rho\rangle \in Q(s, \pi_1)\} \subseteq P. & \end{array}$$

This means, by Lemma 4.4 that

$$Q(s, \pi_1;\pi_2) \quad = \quad \{\langle t, \pi_2\rangle \mid t \in Q(s, \pi_1)\} \cup \{\langle t, \rho;\pi_2\rangle \mid \langle t, \rho\rangle \in Q(s, \pi_1)\}.$$

Hence, for $\pi_1 = pt(\sigma_1)$, $\pi_2 = pt(\sigma_2)$ and $\rho = pt(\tau)$ (which is always the case as pt is an isomorphism). We have

$$\begin{array}{rcl} Q(s, pt(\sigma_1);pt(\sigma_2)) & = & \{\langle t, pt(\sigma_2)\rangle \mid t \in Q(s, pt(\sigma_1))\} \\ & \cup & \{\langle t, pt(\tau);pt(\sigma_2)\rangle \mid \langle t, pt(\tau)\rangle \in Q(s, pt(\sigma_1))\} \end{array}$$

On the other hand we have also

$$\begin{array}{ll} Q(s, pt(\sigma_1;\sigma_2)) & \\ \quad = \{t \mid t \in \sigma_1;\sigma_2(s)\} \cup \{\langle t, pt(\tau)\rangle \mid \langle t, \tau\rangle \in \sigma_1;\sigma_2(s)\} & [\text{Corollary 6.7}] \\ \quad = \{\langle t, pt(\sigma_2)\rangle \mid t \in \sigma_1(s)\} \cup \{\langle t, pt(\tau;\sigma_2\rangle \mid \langle t, \tau\rangle \in \sigma_1(s)\} & [\text{Definition `;'}] \\ \quad = \{\langle t, pt(\sigma_2)\rangle \mid t \in Q(s, pt(\sigma_1))\} & \\ \quad\quad \cup \{\langle t, pt(\tau;\sigma_2)\rangle \mid \langle t, pt(\tau)\rangle \in Q(s, pt(\sigma_1))\}. & [\text{Corollary 6.7}] \end{array}$$

It is now a simple matter to verify that for a fixed $\sigma_2 \in \Sigma_2$

$$d(Q(s, pt(\sigma_1);pt(\sigma_2)), Q(s, pt(\sigma_1;\sigma_2))) \quad \leq \quad \tfrac{1}{2} d(pt(-);pt(\sigma_2), pt(-;\sigma_2)).$$

Therefore we can conclude that

$$\begin{array}{l} d(pt(-;\sigma_2), pt(-);pt(\sigma_2)) \\ \quad = \sup\{d(pt(\sigma_1;\sigma_2), pt(\sigma_1);pt(\sigma_2) \mid \sigma_1 \in \Sigma_2\} \\ \quad = \sup\{d(Q(s, pt(\sigma_1;\sigma_2)), Q(s, pt(\sigma_1);pt(\sigma_2))) \mid \sigma_1 \in \Sigma_2 \;\&\; s \in \mathcal{S}\} \\ \quad \leq \sup\{\tfrac{1}{2} d(pt(-;\sigma_2), pt(-);pt(\sigma_2)) \mid \sigma_1 \in \Sigma_2\} \\ \quad = \tfrac{1}{2} d(pt(-;\sigma_2), pt(-);pt(\sigma_2)). \end{array}$$

$(ParSeq1)$	$S_1 \parallel S_2 \trianglelefteq S_1;S_2$
$(ParSeq2)$	$(S_1;S_2) \parallel S_3 \trianglelefteq S_1;(S_2 \parallel S_3)$
$(Inter1)$	$(A_1;A_2)\Box(A_2;A_1) \trianglelefteq A_1 \parallel A_2$
$(Inter2)$	$(A_1;(S \parallel A_2))\Box(A_2;(A_1;S)) \trianglelefteq (A_1;S) \parallel A_2$
$(Inter3)$	$(A_1;(S_1 \parallel (A_2;S_2))\Box(A_2;((A_1;S_1) \parallel S_2)) \trianglelefteq (A_1;S_1) \parallel (A_2;S_2)$

Figure 4: Some refinement rules for the merge operator in $\mathcal{L}_2$

The above implies $pt(\sigma_1;\sigma_2) = pt(\sigma_1);pt(\sigma_2)$.

(ii) We have

$$\begin{aligned}
& pt(\sigma_1 \Box \sigma_2)(P) \\
&\quad = \{s \in \mathcal{S} \mid (t \in \sigma_1 \Box \sigma_2(s) \Rightarrow t \in P) \;\&\; (\langle t, \tau\rangle \in \sigma_1 \Box \sigma_2(s) \Rightarrow \langle t, pt(\tau)\rangle \in P)\} \\
&\quad = \{s \in \mathcal{S} \mid (t \in \sigma_1(s) \Rightarrow t \in P) \;\&\; (\langle t, \tau\rangle \in \sigma_1(s) \Rightarrow \langle t, pt(\tau)\rangle \in P)\} \\
&\qquad \cap \{s \in \mathcal{S} \mid (t \in \sigma_2(s) \Rightarrow t \in P) \;\&\; (\langle t, \tau\rangle \in \sigma_2(s) \Rightarrow \langle t, pt(\tau)\rangle \in P)\} \\
&\quad = pt(\sigma_1)(P) \cap pt(\sigma_2)(P) \\
&\quad = pt(\sigma_1) \Box pt(\sigma_2)(P).
\end{aligned}$$

(iii) The proof goes along the lines of that of point (i) and hence omitted. $\Box$

As promised at the beginning of this section, we come to the relation between refinement and simulation. A few steps are still necessary. First notice that the domain Σ_2 can be seen as the transition system $\rightarrow \subseteq (\mathcal{S} \times \Sigma_2) \times ((\mathcal{S} \times \Sigma_2) \cup \mathcal{S})$, where

$$\begin{array}{lll}
\langle s, \sigma\rangle \rightarrow t & \text{if} & t \in \sigma(s) \\
\langle s, \sigma\rangle \rightarrow \langle t, \tau\rangle & \text{if} & \langle t, \tau\rangle \in \sigma(s)
\end{array}$$

This is an instance of the so-called 'processes as terms' paradigm [Rut92]. For Σ_2, interpreted as transition system, the notion of (backward) simulation is given in the following definition.

6.12. Definition. A relation $R \subseteq \Sigma_2 \times \Sigma_2$ is called a (backward) simulation if for all $(\sigma, \tau) \in R$ and for all $s \in \mathcal{S}$

(i) if $t \in \tau(s)$ then $t \in \sigma(s)$,

(ii) if $\langle t, \tau'\rangle \in \tau(s)$ then there exists $\langle t, \sigma'\rangle \in \sigma(s)$ such that $(\sigma', \tau') \in R$.

We say that a state transformer σ simulates τ - denoted by $\sigma \trianglelefteq \tau$ - if there exists a simulation relation R with $(\sigma, \tau) \in R$.

We refer to the paper [LV93] for a survey on simulation for transition systems. Intuitively, a state transformer σ simulates τ if for all input states s every computation of $\tau(s)$ can be performed by $\sigma(s)$, preserving all the output states of the atomic steps of the computation. Notice that this implies not only that all the interleaving states are preserved, but also the branching structure. Of special interest for concurrency is the notion of bisimulation for transition systems [Par81]. If we look at (backward) simulation as recursively defined superset order, then bisimulation is recursively defined equality.

6.13. Definition. A relation $R \subseteq \Sigma_2 \times \Sigma_2$ is called a (strong) bisimulation if for all $(\sigma, \tau) \in R$ and for all $s \in \mathcal{S}$

(i) $t \in \tau(s)$ if and only if $t \in \sigma(s)$;

(ii) if $\langle t, \tau'\rangle \in \tau(s)$ then there exists $\langle t, \sigma'\rangle \in \sigma(s)$ such that $(\sigma', \tau') \in R$;

(iii) if $\langle t, \sigma'\rangle \in \sigma(s)$ then there exists $\langle t, \tau'\rangle \in \tau(s)$ such that $(\sigma', \tau') \in R$.

Two state transformers σ and τ are strongly bisimilar —denoted by $\sigma \underline{\leftrightarrow} \tau$— if there exists a bisimulation relation R such that $(\sigma, \tau) \in R$.

Clearly, if $\sigma \underline{\leftrightarrow} \tau$ then $\sigma \trianglelefteq \tau$ and $\tau \trianglelefteq \sigma$. The converse however does not hold, as is shown in the following example (due to J.J.M.M. Rutten). Consider the following functions in Σ_2:

$$\delta = \lambda s.\,\emptyset \qquad \sigma' = \lambda s.\,\{\langle s, \delta\rangle\}$$
$$\sigma = \lambda s.\,\{\langle s, \sigma'\rangle\} \qquad \tau = \lambda s.\,\{\langle s, \sigma'\rangle, \langle s, \delta\rangle\}$$

Define $R = \{(\delta,\delta), (\sigma',\sigma'), (\sigma',\delta), (\sigma,\tau), (\tau,\sigma)\}$. It is not hard to see that R is a simulation relation. Hence $\sigma \trianglelefteq \tau$ and $\tau \trianglelefteq \sigma$. However $\sigma \neq \tau$ and hence by Theorem 6.14 (ii) they are not bisimilar.

We are now ready for stating the relationships between simulation and refinement. Though expressed in two different domains, the two notions coincide. The following theorem gives also a representation of bisimulation as equality in the domain Σ_2. Since Π_2 is isomorphic to Σ_2 we have, as corollary, a representation of bisimulation for predicate transformers in Π_2. As far as we know, this is novel in the context of predicate transformers.

6.14. Theorem. *Let $\sigma, \tau \in \Sigma_2$. Then*
(i) *$\sigma \trianglelefteq \tau$ if and only if $pt(\sigma) \trianglelefteq pt(\tau)$;*
(ii) *$\sigma \underline{\leftrightarrow} \tau$ if and only if $\sigma = \tau$;*
(iii) *$\sigma \underline{\leftrightarrow} \tau$ if and only if $pt(\sigma) = pt(\tau)$.*

Proof. (i) Suppose $\sigma \trianglelefteq \tau$. Then there exists a simulation relation R on Σ_2 such that $(\sigma, \tau) \in R$. Define the relation $\hat{R}$ on Π_2 by
$$(\pi, \rho) \in \hat{R} \quad \Leftrightarrow \quad (st(\pi), st(\rho)) \in R.$$
We need to prove that $\hat{R}$ is a refinement relation. Since pt and st form an isomorphism, we have that$pt(\sigma) \trianglelefteq pt(\tau)$.

Let $(\pi, \rho) \in \hat{R}$ and $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times \Pi_2)$. If $s \in \pi(P)$ then $Q(s, \pi) \subseteq P$ by Lemma 4.4. In order to establish $s \in \rho(\{t \in \mathcal{S} \mid t \in P\} \cup \{\langle t, \rho'\rangle \mid \exists \langle t, \pi'\rangle \in P.\ (\pi', \rho') \in \hat{R}\})$ we need to prove
$$Q(s, \rho) \quad \subseteq \quad \{t \in \mathcal{S} \mid t \in P\} \cup \{\langle t, \rho'\rangle \mid \exists \langle t, \pi'\rangle \in P.\ (\pi', \rho') \in \hat{R}\}$$
If $t \in Q(s, \rho)$ then $t \in st(\rho)$ and hence $t \in st(\pi)$ because $(st(\pi), st(\rho)) \in R$ and R is a simulation relation. But then $t \in Q(s, \pi) \subseteq P$ and hence $t \in \{t \in \mathcal{S} \mid t \in P\}$. If $\langle t, \rho'\rangle \in Q(s, \rho)$ then $\langle t, \tau\rangle \in st(\rho)$ with $\tau = st(\rho')$. Since $(st(\pi), st(\rho)) \in R$ then there exists $\langle t, \sigma\rangle \in st(\pi)$ such that $(\sigma, \tau) \in R$. Hence there exists $\langle t, \pi'\rangle \in Q(s, \pi) \subseteq P$ with $\sigma = st(\pi')$ and $(\pi', \rho') \in \hat{R}$, that is $\langle t, \rho'\rangle \in \{\langle t, \rho'\rangle \mid \exists \langle t, \pi'\rangle \in P.\ (\pi', \rho') \in \hat{R}\}$.

Conversely, suppose $pt(\sigma) \trianglelefteq pt(\tau)$. Then there exists a refinement relation R on Π_2 such that $(pt(\sigma), pt(\tau)) \in R$. Define the relation $\hat{R}$ on Σ_2 by
$$(\sigma, \tau) \in \hat{R} \quad \Leftrightarrow \quad (pt(\sigma), pt(\tau)) \in R.$$
We need to prove that $\hat{R}$ is a simulation relation from which it follows that $\sigma \trianglelefteq \tau$. Let $s \in \mathcal{S}$. Then $s \in pt(\sigma)(Q(s, pt(\sigma)))$ and since $(pt(\sigma), pt(\tau)) \in R$ we have that
$$s \in pt(\tau)(\{t \in \mathcal{S} \mid t \in Q(s, pt(\sigma))\} \cup \{\langle t, \rho'\rangle \mid \exists \langle t, \pi'\rangle \in Q(s, pt(\sigma)).\ (\pi', \rho') \in R\}).$$
Therefore, if $t \in \tau(s)$ then by definition of pt we have that $t \in \{t \in \mathcal{S} \mid t \in Q(s, pt(\sigma))\}$ and hence $t \in Q(s, pt(\sigma))$, that is $t \in \sigma(s)$. If $\langle pt, \tau'\rangle \in \tau(s)$ then $\langle t, pt(\tau')\rangle \in \{\langle t, \rho'\rangle \mid \exists \langle t, \pi'\rangle \in Q(s, pt(\sigma)).\ (\pi', \rho') \in R\}$. Hence there exists $\langle t, \pi'\rangle \in Q(s, pt(\sigma))$ such that $(\pi', pt(\tau')) \in R$. But this means that there exists $\langle t, \sigma'\rangle \in \sigma(s)$ with $pt(\sigma') = \pi'$ and hence $(\sigma', \tau') \in \hat{R}$.

(ii) Similar to [GR89] and hence omitted.

(iii) Immediate by (iii) and Theorems 6.10 and 6.16. □

Using Theorem 6.10, Definition 6.12 and Theorem 6.14 one can straightforwardly prove the correctness of the refinement rules for the language $\mathcal{L}_2$ given in Figures 3 and 4. As expected, we have for example that the sequential composition ';' is associative, the choice '□' is commutative, idempotent and associative, and that the merge '$\|$' is commutative and associative.

$$
\begin{array}{ll}
(CompSeq) & \dfrac{S_1 \trianglelefteq S_1' \ \& \ S_2 \trianglelefteq S_2'}{S_1;S_2 \trianglelefteq S_1';S_2'} \\
(CompBox) & \dfrac{S_1 \trianglelefteq S_1' \ \& \ S_2 \trianglelefteq S_2'}{S_1 \Box S_2 \trianglelefteq S_1' \Box S_2'} \\
(CompPar) & \dfrac{S_1 \trianglelefteq S_1' \ \& \ S_2 \trianglelefteq S_2'}{S_1 \| S_2 \trianglelefteq S_1' \| S_2'}
\end{array}
$$

Figure 5: Some compositional refinement rules for $\mathcal{L}_2$

In Figure 5 we state the compositionality of refinement in $\mathcal{L}_2$ with respect to sequential composition, choice and merge. The rules are not hard to check, once the following observation is made. Let R_1 and R_2 be two simulation relation on Σ_2. Then

$$
\begin{array}{rcl}
R_1;R_2 & = & \{(\sigma_1;\sigma_2, \tau_1;\tau_2) \mid (\sigma_1, \tau_1) \in R_1 \ \& \ (\sigma_2, \tau_2) \in R_2\} \\
R_1 \Box R_2 & = & \{(\sigma_1 \Box \sigma_2, \tau_1 \Box \tau_2) \mid (\sigma_1, \tau_1) \in R_1 \ \& \ (\sigma_2, \tau_2) \in R_2\} \\
R_1 \| R_2 & = & \{(\sigma_1 \| \sigma_2, \tau_1 \| \tau_2) \mid (\sigma_1, \tau_1) \in R_1 \ \& \ (\sigma_2, \tau_2) \in R_2\}
\end{array}
$$

are also refinement relations on Σ_2.

We close this section with the relationship between Π_3 and the complete metric space Σ_3, where Σ_3 is the unique (up to isometry) solution of the domain equation $X = \mathcal{S} \to \mathcal{P}_{cl}(\mathcal{S} + \mathcal{S} \times \frac{1}{2}(X))$ for a fixed set of states $\mathcal{S}$. Also this domain comes equipped with operations similar to those on Σ_2. A main difference, however, is that we need to apply a closure operator to their definitions in order to ensure well-definedness. See, for a discussion, Section 6.3 in [Bre94].

6.15. Definition. Let I be a non empty index set. Define the operators ';', '$\Box$'$_I$ and '$\|$' on Σ_3 by

$$
\begin{array}{rcl}
(\sigma_1;\sigma_2)(s) & = & cl(\{\langle t, \sigma_2 \rangle \mid t \in \sigma_1(s)\} \cup \{\langle t, \tau;\sigma_2 \rangle \mid \langle t, \tau \rangle \in \sigma_1(s)\}) \\
(\Box_I\{\sigma_i \mid i \in I\})(s) & = & cl(\bigcup\{\sigma_i(s) \mid i \in I\}) \\
(\sigma_1 \| \sigma_2)(s) & = & cl(\{\langle t, \sigma_2 \rangle \mid t \in \sigma_1(s)\} \cup \{\langle t, \tau \| \sigma_2 \rangle \mid \langle t, \tau \rangle \in \sigma_1(s)\}) \cup \\
 & & cl(\{\langle t, \sigma_1 \rangle \mid t \in \sigma_2(s)\} \cup \{\langle t, \sigma_1 \| \tau \rangle \mid \langle t, \tau \rangle \in \sigma_2(s)\})
\end{array}
$$

where $\sigma_1, \sigma_2, \sigma_i \in \Sigma_3$ for every $i \in I$ and $s \in \mathcal{S}$.

In similar way as we have seen for the other predicate and state transformer domains, the following theorem states an isomorphism between Σ_3 and Π_3.

6.16. Theorem. *Let $pt : \Sigma_3 \to \Pi_3$ and $st : \Pi_3 \to \Sigma_3$ given by*

$$
\begin{array}{rcl}
pt(\sigma)(P) & = & \{s \in \mathcal{S} \mid (t \in \sigma(s) \Rightarrow t \in P) \ \& \ (\langle t, \tau \rangle \in \sigma(s) \Rightarrow \langle t, pt(\tau) \rangle \in P)\}, \\
st(\pi)(s) & = & \{t \in \mathcal{S} \mid t \in Q(s, \pi)\} \cup \{\langle t, st(\rho) \rangle \in \mathcal{S} \times \Sigma_2 \mid \langle t, \rho \rangle \in Q(s, \pi)\}.
\end{array}
$$

for all $\sigma \in \Sigma_3$, $P \in \mathcal{P}(\mathcal{S} + \mathcal{S} \times Pi_3)$, $\pi \in \Pi_3$, and $s \in \mathcal{S}$. Both pt and st are well defined and form an isometry between Σ_3 and Π_3.

Proof. The proof of the well definedness is similar to those of Lemma 6.6 and Lemma 6.9. The proof of the isometry goes along the lines of Theorem 6.10. $\Box$

Also pt and st preserve the sequential composition, unbounded choice, but we need to be more careful due to the presence of a closure operator in the definition of these operations. The following lemma will be useful for this purpose.

6.17. Lemma. *Let* (X,d) *be a metric space,* $A \in \mathcal{P}(X)$*, and* $Y, C \in \mathcal{C}(X)$*.*

$$(\forall x \in Y.\ x \in cl(A) \Rightarrow x \in C) \quad \Leftrightarrow \quad (\forall x \in Y.\ x \in A \Rightarrow x \in C)$$

Proof. From left to right is simple because $A \subseteq cl(A)$. To prove the other direction pick $x \in cl(A) \cap Y$. Then by definition of closure operator for a metric space there exists a Cauchy sequence $(y_i)_i$ in $A \cap Y$ such that $x = lim_i y_i$. But $y_i \in A \cap Y$ implies $y_i \in C$ for all $i \geq 0$. Since C is closed we have that $x = lim_i y_i \in C$. $\square$

In the following we treat only the unbounded choice. Sequential composition and merge can be treated in a similar way.

6.18. Lemma. *Let* I *be a non-empty index set. Then*
(i) $pt(\sigma_1;\sigma_2) = pt(\sigma_1);pt(\sigma_2)$,
(ii) $pt(\Box_I\{\sigma_i \mid i \in I\}) = \Box_I\{pt(\sigma_i) \mid i \in I\}$,
(iii) $pt(\sigma_1 \| \sigma_2) = pt(\sigma_1) \| pt(\sigma_2)$,
where $\sigma_1, \sigma_2, \sigma_i \in \Sigma_3$ *for all* $i \in I$*.*

Proof. We prove only the second item. The other two can be treated in a similar way. We have for a closed set $P \in \mathcal{C}(\mathcal{S} + \mathcal{S} \times \Pi_3)$

$$\begin{aligned}
&pt(\Box_I\{\sigma_i \mid i \in I\})(P) \\
&= \{s \in \mathcal{S} \mid (t \in cl(\textstyle\bigcup\{\sigma_i(s) \mid i \in I\}) \Rightarrow t \in P) \\
&\quad \& \ (\langle t, \tau\rangle \in cl(\textstyle\bigcup\{\sigma_i(s) \mid i \in I\}) \Rightarrow \langle t, pt(\tau)\rangle \in P)\} \\
&= \{s \in \mathcal{S} \mid (t \in \textstyle\bigcup\{\sigma_i(s) \mid i \in I\} \Rightarrow t \in P) \\
&\quad \& \ (\langle t, \tau\rangle \in \textstyle\bigcup\{\sigma_i(s) \mid i \in I\} \Rightarrow \langle t, pt(\tau)\rangle \in P)\} \qquad [P, \mathcal{S}, \mathcal{S} \times \Pi_3 \text{ closed, Lemma 6.17}] \\
&= \textstyle\bigcap_I\{\{s \in \mathcal{S} \mid (t \in \sigma_i(s) \Rightarrow t \in P) \ \& \ (\langle t, \tau\rangle \in \sigma_i(s) \Rightarrow \langle t, pt(\tau)\rangle \in P)\} \mid i \in I\} \\
&= \textstyle\bigcap_I\{pt(\sigma_i) \mid i \in I\} \\
&= \Box_I\{pt(\sigma_i) \mid i \in I\}.
\end{aligned}$$
$\square$

Also the domain Σ_3 induces a transition system in the same way as for the domain Σ_2. Therefore simulation and bisimulation can be defined exactly as in Definitions 6.12 and 6.13. A counterpart of Theorem 6.5 holds for Σ_3 and Π_3. Due to the closure operator used in Definition 6.15, simulation is, in general, *not* a congruence for ';', '$\Box_I$' and '$\|$'. One could remedy this problem by using a notion of bisimulation up to closure operator. However, since this goes beyond the scope of the present paper, the matter remains as further work.

7 Concluding Remarks

In this paper we have proposed a compositional weakest precondition semantics for two parallel languages with recursion and —finite and unbounded— non-determinacy. The semantics are based on two new metric resumption domains obtained from domain equations involving functors for 'observable' and 'safety' predicate transformers. In these domains a notion of refinement is defined, extending a familiar refinement relation for the sequential non-deterministic case. By establishing a duality between the semantic domains on the one hand and De Bakker-Zucker domains on the other, a direct correspondence, not only between the introduced semantics, but also between the given refinement relation for predicate transformers and the notion of backward simulation for state transformers is obtained. As a corollary, a definition of bisimulation for predicate transformers is derived. A few examples and refinement rules illustrate the proposed models.

A first question to address, facing the four domains Π_i and semantics wp_i, concerns their relative correctness. In work in progress we relate the domains via abstraction mappings. By tuning the abstraction to be applied in the treatment of divergence, we may either obtain a weakest precondition semantics or weakest liberal precondition semantics from the parallel models. Thus both partial and total correctness are represented in wp_2 and wp_3.

With some minor modification strong fairness conditions can be embodied in $\mathcal{L}_3$ and UNITY programs can be expressed in the language. The assertion guards $\{b\}$ have to be added to $\mathcal{L}_3$ (or $\mathcal{L}_2$) as primitive and a subsequent extension of the models is necessary. However, an additional clause involving the operator

$$\{b\}(P) \quad = \quad \{s \mid \mathcal{B}(b)(s) \ \& \ s \in P\}$$

is sufficient. The underlying domain then needs to be adapted as well. Conjunctivity will replace the multiplicativity condition. The dual domains of state transformers need to be extended with a special element to cater for a denotation for *abort*. Locality can be dealt with, using techniques developed in the metric setting. (Cf. [BV95].) It is an open problem, however, how to deal with *angelic* nondeterminacy here.

Complementary to this is the issue of full abstractness or, more restrictedly, of the adequacy of linear models for $\mathcal{L}_2$ and $\mathcal{L}_3$. Since no such feature as message-passing is present in the languages we may expect that a mixture of the 'sequences of pairs'-techniques of [BHR90] and the linear semantics of Lukkien (cf. [Luk91, Luk94]) will provide a more abstract model. Probably the isomorphism of the domains of predicate transformers vs. state transformers may be of help in the comparison.

Acknowledgments We wish to thank Jaco de Bakker for suggesting us to apply the general framework of [BK94a] to a concurrent language. Several discussions with Franck van Breugel, Jan Rutten and Daniele Turi have been very helpful. We appreciate the feedback of the members of the Amsterdam Concurrency Group on a preliminary version of the paper. We finally wish to thank Frits Vaandrager for his kind bibliographic assistance.

References

[Abr87] S. Abramsky. *Domain Theory and the Logic of Observable Properties.* PhD thesis, Queen Mary College, University of London, 1987.

[AR89] P. America and J.J.M.M. Rutten. Solving Reflexive Domain Equations in a Category of Complete Metric Spaces. *Journal of Computer and System Sciences*, 39(3):343–375, 1989.

[Bac80] R.-J.R. Back. *Correctness Preserving Program Refinements: Proof Theory and Applications*, volume 131 of *Mathematical Centre Tracts.* Centre for Mathematics and Computer Science, Amsterdam, 1980.

[Bac81] R.-J.R. Back. On Correct Refinement of Programs. *Journal of Computer and System Sciences*, 23(1):49–68, 1981.

[Ban22] S. Banach. Sur les Operations dans les Ensembles Abstraits et leurs Applications aux Equations Integrales. *Fundamentae Mathematicae*, 3:133–181, 1922.

[Bes89] E. Best. Towards Compositional Predicate Transformer Semantics for Concurrent Programs. In *J.W. de Bakker, 25 jaar Semantiek*, pages 111–117, Amsterdam, 1989.

[BHR90] J.W. de Bakker, E. Horita, and J.J.M.M. Rutten. Fully Abstract Models for Nonuniform Concurrent Languages. *Information and Computation*, ??:???–???, 1990. To appear.

[BJK95] M.M. Bonsangue, B. Jacobs, and J.N. Kok. Duality beyond Sober Spaces: Topological Spaces and Observation Frames. *Theoretical Computer Science*, ??:???–???, 1995.

[BK94a] M.M. Bonsangue and J.N. Kok. Relating Multifunctions and Predicate Transformers through Closure Operators. In Masami Hagiya and John C. Mitchell, editors, *TACS '94, Sendai, Japan*, volume 789 of *Lecture Notes in Computer Science*, pages

822–843. Springer-Verlag, 1994. Available through anonymous ftp from ftp.cs.vu.nl as /pub/bonsangue/multifunct.ps.Z.

[BK94b] M.M. Bonsangue and J.N. Kok. The Weakest Precondition Calculus: Recursion and Duality. *Formal Aspects of Computing*, 6A:???–???, 1994.

[BKS88] R.J.R. Back and R. Kurki-Suonio. Distributed Cooperation with Action Systems. *ACM Transactions on Programming Languages and Systems*, 10:513–554, 1988.

[Bre93] F. van Breugel. Relating State Transformation Semantics and Predicate Transformer Semantics for Parallel Programs. Technical Report CS-9339, Centre for Mathematics and Computer Science, Amsterdam, 1993.

[Bre94] F. van Breugel. *Topological Models in Comparative Semantics*. PhD thesis, Vrije Universiteit Amsterdam, 1994.

[BS92] R.J.R. Back and K. Sere. Superposition Refinement of Parallel Algorithms. In K.R. Parker and G. Rose, editors, *Formal Description Techniques IV*, pages 475–494. North-Holland, 1992.

[BS94] R.J.R. Back and K. Sere. Action Systems with Synchronous Communication. In E.-R. Olderog, editor, *Programming Concepts, Methods and Calculi*, volume A-56 of *IFIP Transaction*, pages 107–126. Elsevier Science Publishers B.V. North-Holland, 1994.

[BV95] J.W. de Bakker and E. de Vink. *Control Flow Semantics*. MIT Press, 1995. To appear.

[BW90] R.-J.R. Back and J. von Wright. Refinement Calculus, part I: Sequential Nondeterministic Programs. In J.W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Stepwise Refinement of Distributed Systems: Models, Formalisms, Correctness*, volume 430 of *Lecture Notes in Computer Science*, pages 42–66. Springer-Verlag, 1990.

[BZ82] J.W. de Bakker and J.I. Zucker. Processes and the Denotational Semantics of Concurrency. *Information and Control*, 54:70–120, 1982.

[CM88] K.M. Chandy and J. Misra. *A Foundation of Parallel Program Design*. Addison-Wesley, 1988.

[Dij76] E.W. Dijkstra. *A Discipline of Programming*. Prentice-Hall, 1976.

[DS90] E.W. Dijkstra and C.S. Scholten. *Predicate Calculus and Program Semantics*. Springer-Verlag, New York, 1990.

[Dug66] J. Dugundji. *Topology*. Allyn and Bacon, inc., 1966.

[Eng89] R. Engelking. *General Topology*, volume 6 of *Sigma Series in Pure Mathematics*. Heldermann Verlag, Berlin, revised and completed edition, 1989.

[GM91] P. Gardiner and C. Morgan. Data Refinement of Predicate Transformers. *Theoretical Computer Science*, 87:143–162, 1991.

[GR89] R. van Glabbek and J. Rutten. The processes of De Bakker and Zucker represent bisimulation equivalence classes. In *J.W. de Bakker, 25 jaar Semantiek*, pages 243–246, Amsterdam, 1989.

[Hah48] H. Hahn. *Reelle Funktionen*, volume 1: Punktfuntionen. Chelsea Publishing Company, New York, 1948.

[Hes88a] W.H. Hesselink. An Algebraic Calculus of Command. Technical Report CS-8808, Rijksuniversiteit Groningen, Department of Mathematics and Computing Science, 1988.

[Hes88b] W.H. Hesselink. Deadlock and fairness in morphisms of transition systems. *Theoretical Computer Science*, 59:235–257, 1988.

[Hes89] W.H. Hesselink. Predicate Transformer Semantics of General Recursion. *Acta Informatica*, 26:309–332, 1989.

[HP72] P. Hitchcock and D. Park. Induction Rules and Termination Proofs. In M. Nivat, editor, *Proc. 1st ICALP*, Rocquencourt, France, 1972. North-Holland.

[KR90] J.N. Kok and J.J.M.M. Rutten. Contractions in Comparing Concurrency Semantics. *Theoretical Computer Science*, 76 (2/3):179–222, 1990.

[Kur56] K. Kuratowski. Sur une Methode de Metrisation Complete des Certain Espaces d'Ensembles Compacts. *Fund. Math.*, 42:114, 1956.

[Kwi91] M.Z. Kwiatowska. On Topological Characterization of Behavioural Properties. In G.M. Reed, A.W. Roscoe, and R.F. Wachter, editors, *Topology and Category Theory in Computer Sciences - Proc. Oxford Topology Symposioum, June 1990*, pages 153–177. Oxford Science Publications, 1991.

[Lam90] L. Lamport. win and sin: Predicate Transformers for Concurrency. *ACM Transaction on Programming Languages and Systems*, 12(3):396–428, 1990.

[Luk91] J.J. Lukkien. *Parallel Program Design and Generalized Weakest Precondition.* PhD thesis, Rijksuniversiteit Groningen, 1991.

[Luk94] J.J. Lukkien. Operational Semantics and Generalized Weakest Precondition. *Science of Computer Programming*, 22:137–155, 1994.

[LV93] N.A. Lynch and F.W. Vaandrager. Forward and Backward Simulation - part I: Untimed Systems. Technical Report CS-R9313, Centre for Mathematics and Computer Science, Amsterdam, 1993. To appear in Information and Computation.

[Mor87] J. Morris. A Theoretical Basis for Stepwise Refinement and the Programming Calculus. *Science of Computer Programming*, 9:287–306, 1987.

[Par81] D.M.R. Park. Concurrency and Automata on Infinite Sequences. In P. Deussen, editor, *Proc. 5th GI Conference*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer-Verlag, 1981.

[Plo79] G.D. Plotkin. Dijkstra's Predicate Transformer and Smyth's Powerdomain. In D. Bjorner, editor, *Proc. of the Winter School on Abstract Software Specification*, volume 86 of *Lecture Notes in Computer Science*, pages 527–553. Springer-Verlag, 1979.

[RT93] J.J.M.M. Rutten and D. Turi. On the Foundations of Final Semantics: Non-Standard Sets, Metric Spaces, Partial Orders. In J.W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Rex Workshop '92 'Semantics: Foundations and Applications'*, volume 666 of *Lecture Notes in Computer Science*, pages 477–530. Springer-Verlag, 1993.

[Rut92] J.J.M.M. Rutten. Processes as Terms: Non-Well-Founded Models for Bisimulation. Technical Report CS-R9211, Centre for Mathematics and Computer Science, Amsterdam, 1992. To appear in Mathematical Structures in Computer Science.

[Ser90] K. Sere. *Stepwise Derivation of Parallel Algorithms.* PhD thesis, Åbo Akademi, 1990.

[Smy83] M.B. Smyth. Power Domains and Predicate Transformers: A Topological View. In J. Diaz, editor, *Proc. 10th ICALP*, volume 154 of *Lecture Notes in Computer Science*, pages 662–675, Barcelona, Spain, 1983. Springer-Verlag.

[SZ92] D. Scholefield and H.S.M. Zedan. Weakest Precondition Semantics for Time and Concurrency. *Information Processing Letters*, 43(6):301–308, 1992.

[US93] A. Udaya Shankar. An Introduction to Assertional Reasoning for Concurrent Systems. *ACM Computing Surveys*, 25:225–262, 1993.

[Vic89] S.J. Vickers. *Topology via Logic*, volume 5 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1989.

[Vin90] E.P. de Vink. Comparative semantics for Prolog with cut. *Science of Computer Programming*, 13:237–264, 1989/90.

[Wri90] J. von Wright. *A Lattice-theoretical Basis for Program Refinement*. PhD thesis, Åbo Akademi, 1990.