# STRICTNESS ANALYSIS FOR POLYREC, A LANGUAGE WITH POLYMORPHIC AND RECURSIVE TYPES

Gerard R. Renardel de Lavalette

Department of Philosophy

University of Utrecht

Logic Group
Preprint Series
No. 33

Department of Philosophy

University of Utrecht

# Strictness analysis for POLYREC, a language with polymorphic and recursive types

Gerard R. Renardel de Lavalette

*University of Utrecht, Department of Philosophy*
*Heidelberglaan 2, 3584 CS Utrecht, The Netherlands*

Abstract

In this paper we define the functional language POLYREC with polymorphic and recursive types, and develop concrete and abstract interpretations for this language. These semantics, based on complete partial orderings and complete lattices, are subsequently used for definition and analysis of strictness.

1980 Mathematical Subject Classification (1985): 03B40, 68N15
1986 CR Classification System: D.3.1, F.3.1, F.3.2, F.4.1

Key words and phrases: strictness analysis, functional language, recursive types, polymorphic lambda calculus, abstract interpretation.

Note: This work was supported by the Dutch Parallel Reduction Machine Project, sponsored by the Dutch Ministry of Science & Education (Science Council).

March 1988

**CONTENTS.**



## §1. INTRODUCTION.

### 1.1. Strictness analysis.

The implementation of functional languages provides an interesting testing ground for probing the (sometimes high-running) expectations on the benefits of parallellism, and is pursued by several research groups. One of these is the Dutch Parallel Reduction Machine Project, which sponsored the research for this paper.
We start with a simplified picture of the situation. A functional program can be seen as a term; evaluation of the program comes down to reducing that term according to a set of rewrite rules, until no rule can be applied and a normal form is reached. A reduction step consists of replacing a subterm by its reduct, i.e. the result of applying a rewrite rule. Now these reduction steps can be made in successive order, but this is not necessarily so: it seems reasonable to expect that executing several reductions simultaneously will reduce the time needed to obtain a normal form.
However, the possible consequence of evaluating different subterms in parallel is that subterms are evaluated the outcome of which is not needed for the result of the main term: consider e.g.

T := if A then B else C;

evaluating A, B and C in parallel means doing more than is strictly necessary, since either B (if A is false) or C (if A is true) is not needed for the evaluation of T.
So it seems reasonable only to evaluate those subterms which are needed for the final result, where a subterm T (not in normal form) of a term S is called *needed* if every reduction of S to normal

form contains a contraction within T. This definition of neededness is operational and not very manageable, therefore it is often replaced by the semantical notion of strictness: a term T is called *strict* if $[T]\perp = \perp$, where [.] is some interpretation, and a subterm T of a term S=S(T) is called *strict* if $\lambda x.S(x)$ is strict. In other words: T needs to be defined for S(T) to be meaningful. For the untyped lambda calculus the equivalence

every redex R is head needed in TR $\Leftrightarrow$ T is strict

(where head needed means: needed in any reduction to head normal form) has been proved by Hans Mulder: see [BKKS86], 5.1. Analysis of the situation for the rewrite system of our language POLYREC is an object of future research.
Now, if we restrict our strategy of evaluating in parallel to: evaluate only strict subterms, we know that all we do is necessary for the final result. This strategy of strict parallel evaluation can be implemented for a functional language once its semantics is given: but it requires the evaluation of the interpretation of terms, which is as difficult (if not the same) as reduction to normal form. So for this strategy to be efficient, we cannot base it on an exact computation of strictness behaviour but only on an approximation, which we call *demonstrable* strictness. This is strictness based on a simplification of the semantics called abstract interpretation, which is essentially easier to evaluate than the concrete interpretation.

1.2. Underlying ideas.

We now sketch the ideas behind our approach in somewhat more detail. POLYREC is the result of devising a functional language with the following features:

typed lambda calculus;
arithmetical and logical operators;
recursively defined terms (using a fixed point operator);
polymorphic terms (using type abstraction);
operations on types (product, sum, exponent);
recursively defined types (using a fixed point operator on type functions).

There are four kinds of expressions of POLYREC: types, type constructors, terms and polymorphic terms. In the interpretations of POLYREC defined in this paper, this is what happens with these kinds:

types are mapped on objects of a category,
type constructors are mapped on functors,
terms are mapped on elements of objects,
polymorphic terms are mapped on choice functions associated with functors.

The first idea for the concrete interpretation CI is to interpret the types of POLYREC by complete partial orderings, since they allow the definition of fixed point operators and are closed under disjoint sum, product, exponentiation (=taking continuous function spaces) and taking projective limits (used to interpret recursive types, defined as the least fixed point of type constructors). With such an interpretation CI, strictness is defined by

a term t of type $\sigma\to\tau$ is *strict* if $CI(t)\bot = \bot$

It is our goal to appoximate this by the notion of d(emonstrably)-strict, defined by

a term t of type $\sigma\to\tau$ is *d-strict* if $AI(t)\bot = \bot$

using an abstract interpretation AI. We want this to be an approximation from above, i.e.

if t is d-strict, then t is strict,

and this requires

(1) $AI(t)\bot = \bot \Rightarrow CI(t)\bot = \bot$.

There is, of course, a trivial solution to this: define AI in such a way that it never yields $\bot$, then no term is d-strict. To obtain a more interesting abstract interpretation, we proceed as follows. First we consider the interpretation of constant types, e.g. N, the type of the natural numbers. We have $CI(N) = N = <\{\bot,0,1,2,...\},\leq>$ where $x\leq y$ iff$(x=\bot$ or $x=y)$. We can abstract from this structure by identifying all natural numbers to one object $\top$ : this yields the cpo $\underline{2} = <\{\bot,\top\},\leq>$ with $\bot\leq\top$, and the mapping $abs_N: N \to \underline{2}$ defined by $abs(x) = \bot$ if $x=\bot$, $\top$ otherwise. The same can be done for other type constants such as B (the type of the booleans).
The idea now is: extend abs to all types of the language, then use it to define AI by putting something like

$AI(c) := abs(CI(c))$ for c a constant
$AI(st) := AI(s)AI(t)$
$AI(\lambda x.t) := \lambda a.AI(t[x:=a])$

and prove (1) by showing

(2) $abs \circ CI \leq AI$ and $abs(x)=\bot \leftrightarrow x=\bot$

(2) is proved with induction over types and terms: the main lemma is

$$abs(fd) \leq abs(f)abs(d)$$

to satisfy this, abs(f) is defined (after [BHA86]) by

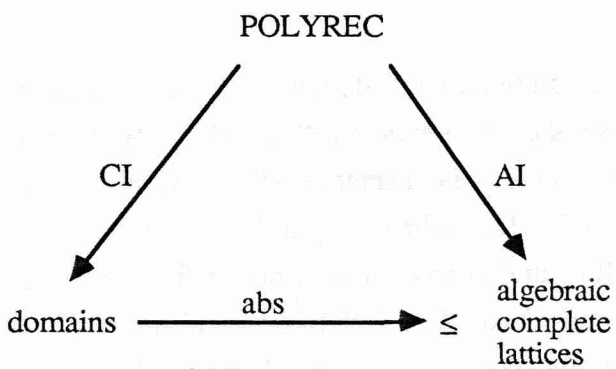
$$abs(f)a := \bigvee\{abs(fd) \mid abs(d) \leq a\}:$$

as $\{abs(fd) \mid abs(d) \leq a\}$ is in general not directed, this definition is not correct in an arbitrary cpo, and we are led to *complete lattices*, where every subset has a supremum, as the realm for the abstract interpretation. Another restriction is imposed by the fact that we need

$$f \text{ continuous} \rightarrow abs(f) \text{ continuous},$$

and this can be shown if the cpo's involved are *algebraic*.
So on the one (concrete) hand we have algebraic cpo's; to guarantee closure under $\rightarrow$ , they also have to be *boundedly complete*. Such cpo's (being both algebraic and boundedly complete) are called *domains*. On the other (abstract) hand, we have algebraic complete lattices, *acl*'s for short.

So the situation can be rendered as follows:




Here the $\leq$ symbol near the arrow for abs indicates $abs \circ CI \leq AI$, the first part of (2).

The thing to do now is to construct, for every type $\tau$ of the language POLYREC, a domain $CI(\tau)$ and an algebraic complete lattice $AI(\tau)$. This is simple for type constants such as N and B, and for types constructed with $\times$, + and $\rightarrow$; for types $\mu\Phi$, which can be seen as the fixed point of the unary type operator $\Phi$, this is done in the style of [SP82] with a limit construction which does not only involve a sequence of domains resp. acl's, but also continuous functions between them. These functions are obtained by considering $CI(\Phi)$ not only as an operator $\lambda D.CI(\Phi(D))$ on domains, but also as a *functor* F on the category $\mathbb{D}$ of domains, i.e. a mapping defined on both domains and continuous functions between them, satisfying (among other things) $F(f \circ g) = F(f) \circ F(g)$. Moreover, these functors F have to be continuous, i.e. to satisfy $F(\lim_n X_n) =$

$\lim_n(F(X_n))$. The definition of $AI(\Phi)$ proceeds in parallel.
For the definition of $abs(\mu\Phi)$, we use a slightly different limit construction involving a sequence of functions obtained by what we call an *abstraction transformation* between two functors: this is a generalisation of the notion of natural transformation.

1.3. Outline of the rest of this paper.

We sketch the contents of the rest of this paper. In §2 a definition of POLYREC and its axiomatics is given, together with some examples. §3 and §4 form a detailed presentation of complete partial orderings, domains and algebraic complete lattices and their relevant properties. §5 starts with a concise introduction of categories and functors, followed by the definition of four categories to be used in the next sections. The central notions of abstraction - an arrow in the category of domains with certain properties - and abstraction transformation are the subject matter of §6. In §7 several fixpoint constructions using chains of domains are worked out in extenso; they are needed to deal with recursively defined types in §8, where the concrete and the abstract interpretation of POLYREC are given, together with an abstraction function for every type. The harvest is reaped in §9: definition of strictness and demonstrable strictness, and the theorem relating the two. In §10 we look back, discuss some variants on the method and suggest topics for future research. The Appendix contains a justification of the definition of abstraction transformation given in §6.

1.4. Acknowledgements.
The author wishes to thank Samson Abramsky and Chris Hankin for useful discussions, Piet Rodenburg for a crucial insight at a crucial moment, and Henk Barendregt for many stimulating and refreshing meetings concerning the subject of this paper and other things.
Written sources of inspiration were [BHA86] (for the definition of $\twoheadrightarrow$ in 6.5.(i) and the outline of §8), [Ba84], [GL84] and [Lo79] (all for §3).

## § 2. THE LANGUAGE POLYREC.

In this section we define the functional language POLYREC. It is a strongly typed language with recursive and polymorphic types, but strictly weaker than second-order lambda calculus (see Remark 2.4).

2.1. Types.

We define (inductively and simultaneously) the collections $\Theta$ (with elements $\sigma, \tau, ...$) of *types* and $\Theta^n \to \Theta$ (with elements $\Phi, \psi, ...$; n=1,2,...) of *n-ary type constructors*. We assume the collection

TYPvar (with infinitely many elements $\alpha, \beta, ...$)

to be given, and put

$\mathrm{TYPvar} \subseteq \Theta$
$\mathsf{N}, \mathsf{B} \in \Theta$
$\times, +, \to \in \Theta^2 \to \Theta$
if $\Phi \in \Theta^n \to \Theta$ and $\tau_1, ..., \tau_n \in \Theta$, then $\Phi\tau_1...\tau_n \in \Theta$
if $\tau \in \Theta$ and $\alpha_1, ..., \alpha_n \in \mathrm{TYPvar}$, then $\delta\alpha_1...\alpha_n.\tau \in \Theta^n \to \Theta$
if $\Phi \in \Theta \to \Theta$, then $\mu\Phi \in \Theta$.

2.2. Terms.

We define the collection T (with elements s, t,...) of *terms* and $\Theta^n \to T$ (with elements p, q,...; n=1,2,...) of *n-ary polymorphic terms*, together with the functions **type** : $T \to \Theta$ and **type** : $(\Theta^n \to T) \to (\Theta^n \to \Theta)$ (n=1,2,...). We assume the collection

TERMvar (with elements x, y,...), satisfying:
for every $\tau \in \Theta$ there are infinitely many x with **type**(x)=$\tau$,

to be given, and put

$\mathrm{TERMvar} \subseteq T$
$\mathsf{0}, \mathsf{S}, \mathsf{plus}, \mathsf{times}, \mathsf{eq}, \mathsf{true}, \mathsf{false}, \mathsf{not}, \mathsf{and}, \mathsf{or} \in T$
$\bot, \mathsf{ite}, \mathsf{fix} \in \Theta \to T$
$\mathsf{pair}, \mathsf{p0}, \mathsf{p1}, \mathsf{in0}, \mathsf{in1} \in \Theta^2 \to T$
$\mathsf{case} \in \Theta^3 \to T$
if $s, t \in T$ and **type**(s) = **type**(t)$\to\tau$, then $(st) \in T$
if $t \in T$, $x \in \mathrm{TERMvar}$ and **type**(x)=$\tau$, then $\lambda x{:}\tau.t \in T$
if $p \in \Theta^n \to T$ and $\tau_1, ..., \tau_n \in \Theta$, then $p(\tau_1, ..., \tau_n) \in T$

if $t \in T$ and $\alpha_1,...,\alpha_n \in$ TYPvar, then $\Lambda\alpha_1...\alpha_n.t \in \Theta^n \to T$, provided that no type variables occurring in the types of free term variables in t are bound

**type** is defined by

$$
\begin{aligned}
&\textbf{type}(0) && = N\\
&\textbf{type}(S) && = N \to N\\
&\textbf{type}(plus) && = N \times N \to N\\
&\textbf{type}(times) && = N \times N \to N\\
&\textbf{type}(eq) && = N \times N \to B\\
&\textbf{type}(true) && = B\\
&\textbf{type}(false) && = B\\
&\textbf{type}(not) && = B \to B\\
&\textbf{type}(and) && = B \times B \to B\\
&\textbf{type}(or) && = B \times B \to B\\
&\textbf{type}(\bot) && = \delta\alpha.\alpha\\
&\textbf{type}(ite) && = \delta\alpha.B \to \alpha \to \alpha \to \alpha\\
&\textbf{type}(fix) && = \delta\alpha.(\alpha \to \alpha) \to \alpha\\
&\textbf{type}(pair) && = \delta\alpha\beta.\alpha \to \beta \to \alpha \times \beta\\
&\textbf{type}(p0) && = \delta\alpha\beta.\alpha \times \beta \to \alpha\\
&\textbf{type}(p1) && = \delta\alpha\beta.\alpha \times \beta \to \beta\\
&\textbf{type}(in0) && = \delta\alpha\beta.\alpha \to \alpha + \beta\\
&\textbf{type}(in1) && = \delta\alpha\beta.\beta \to \alpha + \beta\\
&\textbf{type}(case) && = \delta\alpha\beta\gamma.(\alpha \to \gamma) \to (\beta \to \gamma) \to (\alpha + \beta) \to \gamma\\
&\textbf{type}(st) && = \tau \text{ if } \textbf{type}(s) = \textbf{type}(t) \to \tau\\
&\textbf{type}(\lambda x{:}\tau.t) && = \tau \to \textbf{type}(t)\\
&\textbf{type}(p(\tau_1,...,\tau_n)) && = \textbf{type}(p)(\tau_1,...,\tau_n)\\
&\textbf{type}(\Lambda\alpha_1...\alpha_n.t) && = \delta\alpha_1...\alpha_n.\textbf{type}(t)
\end{aligned}
$$

Notational convention: we often drop parentheses to improve readability, using associativity of application to the left. So ((rs)t) is written rst or r s t . Types are sometimes dropped if this causes no ambiguity.

2.3. Axioms.

Besides the usual axioms and rules for equality, POLYREC has the following axioms:

$$(\delta\alpha_1...\alpha_n.\tau)\tau_1...\tau_n = \tau[\alpha_1 := \tau_1,...,\alpha_n := \tau_n]$$

$$(\delta\alpha.\tau)(\mu(\delta\alpha.\tau)) = \mu(\delta\alpha.\tau)$$

usual equational axioms for 0, S, plus, times, eq, true, false, not, and, or

$\mathsf{ite}(\tau)\ \mathsf{true}\ s\ t = s$
$\mathsf{ite}(\tau)\ \mathsf{false}\ s\ t = t$
$\mathsf{ite}(\tau)\ \bot(B)\ s\ t = \bot(\tau)$

$t(\mathsf{fix}(\tau)t) = \mathsf{fix}(\tau)t$

$\mathsf{p0}(\sigma,\tau)(\mathsf{pair}(\sigma,\tau)\ s\ t) = s$
$\mathsf{p1}(\sigma,\tau)(\mathsf{pair}(\sigma,\tau)\ s\ t) = t$

$\mathsf{case}(\rho,\sigma,\tau)\ r\ s\ (\mathsf{in0}(\rho,\sigma)t) = rt$
$\mathsf{case}(\rho,\sigma,\tau)\ r\ s\ (\mathsf{in1}(\rho,\sigma)t) = st$

$(\lambda x{:}\mathbf{type}(t).s)t = s[x:=t]$

$(\Lambda\alpha_1\ldots\alpha_n.t)\tau_1\ldots\tau_n = t[\alpha_1:=\tau_1,\ldots,\alpha_n:=\tau_n]$

2.4. Remark.
POLYREC is essentially weaker than $\Lambda^2$, the second-order lambda calculus. In $\Lambda^2$ we have e.g.

$(\Lambda\alpha.\lambda x{:}\alpha.x)(\forall\beta.\beta\to\beta)$ is a term of type $(\forall\beta.\beta\to\beta)\to(\forall\beta.\beta\to\beta)$;

this (reading $\delta$ for $\forall$) is impossible in POLYREC, since $\delta\beta.\beta\to\beta$ is not a type, but a type function.

2.5. Examples.
We show the expressive power of POLYREC by giving some type definitions.
i) Lists of objects of type $\tau$:

$L = L(\tau) := \mu(\delta\alpha.1+(\tau\times\alpha))$,

so $L = 1+(\tau\times L)$. The constant $<>$ and the functions cons, hd, tl and lth with

$\mathbf{type}(<>) = 1$
$\mathbf{type}(\mathrm{cons}) = \tau\to L\to L$
$\mathbf{type}(\mathrm{hd}) = L\to\tau$
$\mathbf{type}(\mathrm{tl}) = L\to L$
$\mathbf{type}(\mathrm{lth}) = L\to N$

are defined by

$<> := \mathsf{in0}\,\bot$ (in full: $\mathsf{in0}(1,\tau\times L)\ \bot(1)$)

$\mathrm{cons} := \lambda xy.(\mathsf{in1}\ (\mathsf{pair}\ x\ y))$

(in full: $\lambda x{:}\tau.\lambda y{:}L.(\mathsf{in1}(1,\tau\times L)\ (\mathsf{pair}(\tau,L)\ x\ y))$)

$\mathrm{hd} := \mathsf{case}\ \lambda x.\bot\ \mathsf{p0}$

(in full: $\mathsf{case}(1,\tau\times L,\tau)\ \lambda x{:}1.\bot(\tau)\ \mathsf{p0}(\tau,L)$)

$\mathrm{tl} := \mathsf{case}\ \lambda x.\bot\ \mathsf{p1}$

(in full: $\mathsf{case}(1,\tau\times L,L)\ \lambda x{:}1.\bot(L)\ \mathsf{p1}(\tau,L)$)

$\mathrm{lth} := \phi(\lambda f.(\mathsf{case}\ \lambda x.0\ \lambda x.S(f(\mathsf{p1}x)))$

(in full: $\phi(L\to N)(\lambda f{:}L\to N.(\mathsf{case}(1,\tau\times L,N)\ \lambda x{:}1.0\ \lambda x{:}\tau\times L.S(f(\mathsf{p1}(\tau,L)x)))))$

ii) A model of the type-free lambda calculus over a type $\tau$:

$$M = M(\tau) := \mu(\delta\alpha.\tau+(\alpha\to\alpha)),$$

so $M = \tau + (M \to M)$, and we can define emb, fun and graph with

$\mathbf{type}(\mathrm{emb}) = \tau \to M$

$\mathbf{type}(\mathrm{fun}) = M \to M \to M$

$\mathbf{type}(\mathrm{graph}) = (M \to M) \to M$

by

$\mathrm{emb} := \mathsf{in0}(\tau,M\times M)$

$\mathrm{fun} := \mathsf{case}\ \lambda x.\bot\ \lambda x.x$

(in full: $\mathsf{case}(\tau,M\to M,M\to M)\ \lambda x{:}\tau.\bot(M\to M)\ \lambda x{:}M\to M.x$)

$\mathrm{graph} := \mathsf{in1}(\tau,M\times M)$

## § 3. COMPLETE PARTIAL ORDERINGS.

In this section we consider several types of complete partial orderings (cpo's) and deduce some properties. Our main goal is to find two classes of cpo's for the two interpretations CI and AI of POLYREC we have in mind: both classes consist of so-called algebraic cpo's, in which every element can be approximated by compact elements, and have to be closed under $\rightarrow$ (taking continuous function spaces). This leads to domains, i.e. cpo's which are both algebraic and boundedly complete. For AI, however, we need something more: existence of suprema of arbitrary subsets, and this brings us to the class of algebraic complete lattices. We end the section with a topological characterisation of domains as $f_0$-spaces.

3.1. Definition.
Let $\langle X,\leq\rangle$ be an ordered set. The collection $D(X)$ of *directed* subsets of X is defined by

$$D(X) := \{Y\subseteq X \mid Y\neq\emptyset \wedge \forall xy\in Y\exists z\in Y(x\leq z \wedge y\leq z)\}.$$

$\langle X,\leq\rangle$ is a *complete partial order (cpo)* iff:
a) $\langle X,\leq\rangle$ is a partial order (i.e. $\leq$ is reflexive, transitive and antisymmetric);
b) X contains a smallest element $\perp$;
c) each $Y\in D(X)$ has a *least upper bound (supremum)* $\bigvee Y$, which satisfies

$$\forall y\in Y(y\leq \bigvee Y) \quad (\bigvee Y \text{ is an upper bound of Y}),$$
$$\forall x\in X(\forall y\in Y(y\leq x) \rightarrow \bigvee Y\leq x) \quad (\bigvee Y \text{ is a least upper bound of Y}).$$

iii) Let $\langle X,\leq\rangle$ be a cpo. The topology $\langle X,\mathbf{O}(X)\rangle$ is defined by

$$\mathbf{O}(X) := \{O\subseteq X \mid \forall x\in O\forall y\in X(x\leq y\rightarrow y\in O) \wedge \forall Y\in D(X)(\bigvee Y\in O\rightarrow Y\cap O\neq\emptyset)\},$$

i.e. O is open iff O is upward closed and its complement is closed with respect to suprema of directed subsets.
We often indicate an ordering $\langle X,\leq\rangle$ or a topology $\langle X,\mathbf{O}(X)\rangle$ simply by X.

3.2. Remarks.
i) This topology is sometimes called the *Scott topology* of X (e.g. in [Ba84]), after D.S.Scott, who introduced it in [Sc76]. Scott called it the *topology of positive information* ([Sc76], p. 524). We refer to these references for a proof that $\langle X,\mathbf{O}(X)\rangle$ is indeed a topology.
ii) We follow [Ba84] in excluding $\emptyset$ from $D(X)$. This does not look very elegant, but it allows for a nice characterisation of continuous functions: see 3.4.(i).
iii) Clause (c) in the definition of cpo's allows for the definition of fixed points: see 4.7.

3.3. Fact.
$\{y \in X \mid \text{not}(y \leq x)\}$ is open, for every $x \in X$.

3.4. Lemma.
Let $X$, $Y$ be cpo's, $f : X \to Y$.
i) $f$ continuous $\Leftrightarrow \forall Z \in D(X)(f(\bigvee Z) = \bigvee f(Z))$.
ii) $f$ continuous $\Rightarrow f$ monotonic.
iii) If $X$ is finite, then: $f$ monotonic $\Leftrightarrow f$ continuous.
Proof.i) As in [Ba84], 1.2.6.
ii) Assume f is continuous. If $x \leq y$ then $\{x,y\}$ is directed and $\bigvee\{x,y\} = y$, so $fy = f\bigvee\{x,y\} = \bigvee f\{x,y\} = \bigvee\{fx,fy\}$, hence $fx \leq fy$.
iii) If $X$ is finite, then every directed subset $Y$ of $X$ has a greatest element $y$, so $f\bigvee Y = fy$; also, by monotonicity, $fy = \bigvee fY$ and we conclude that $f$ is continuous. []

3.5. Definition.
Let $\langle X,\leq\rangle$ and $\langle X',\leq'\rangle$ be ordered sets. The *exponent* $[X \to X'] = \langle X'',\leq''\rangle$ is defined by

$$X'' := [X \to X'] = \{f : X \to X' \mid f \text{ continuous}\}$$
$$f \leq'' g \text{ iff } \forall x \in X(fx \leq' gx).$$

3.6. Lemma.
If $X$ and $Y$ are cpo's then so is $[X \to Y]$.
Proof. One easily checks that $[X \to Y]$ is a partial ordering if $X$ and $Y$ are. Likewise one observes that $\lambda x.\bot_Y$ is the least element of $[X \to Y]$.
As to the least upper bound, we have: if $Z \subseteq [X \to Y]$ is directed, then so is $Zx := \{zx \mid z \in Z\}$ for every $x \in X$, and $\bigvee Z = \lambda x.\bigvee Zx$. []

3.7. Definition.
Let $\langle X,\leq\rangle$ be a cpo, $x \in X$.
i) $c(X)$, the collection of *compact* elements of X, is defined by

$$c(X) := \{x \in X \mid \forall Y \in D(X)(x \leq \bigvee Y \to \exists y \in Y(x \leq y))\}.$$

ii) $\Downarrow_c(x)$, the *cone of compact elements under* x, is defined by

$$\Downarrow_c(x) := \{y \in c(X) \mid y \leq x\}.$$

iii) $\langle X,\leq\rangle$ is called an *algebraic* cpo if every element $x$ is the supremum of its cone $\Downarrow_c(x)$ and this cone is directed, i.e.

$$\forall x\in X(\Downarrow_c(x)\in D(X) \wedge x=\bigvee\Downarrow_c(x)).$$

3.8. Fact.
If $Y\in D(X)$, $x\in c(X)$ and $\bigvee Y=x$, then $x\in Y$.

We now investigate the continuous function space $[X \to Y]$ for algebraic cpo's $X$ and $Y$. By lemma 3.6, it is a cpo; is it again algebraic? The answer will be: no, but it *almost* is. More precisely: every $f\in [X \to Y]$ is the supremum of $\Downarrow_c(f)$, but $\Downarrow_c(f)$ is not always directed.
First we define a subset of the compact functions, the *admissible finite* functions.

3.9. Definition.
Let $X$, $Y$ be algebraic cpo's, and let $\mathbf{a}=a_1,...a_n$ and $\mathbf{b}=b_1,...,b_n$ be finite sequences of compact elements of $X$ resp. $Y$. We define $Adm(\mathbf{a},\mathbf{b})$ by

$$Adm(\mathbf{a},\mathbf{b}) := \forall i,j\leq n((a_i\leq a_j \to b_i\leq b_j ) \wedge \forall x\in X(a_i,a_j\leq x \to \exists k\leq n(a_i,a_j\leq a_k\leq x)))$$

and $f_{\mathbf{ab}}$ by

$$f_{\mathbf{ab}}(x) := \bigvee\{b_i \mid a_i\leq x\}.$$

Functions of the form $f_{\mathbf{ab}}$ are called *finite functions*; if also $Adm(\mathbf{a},\mathbf{b})$ holds, then $f_{\mathbf{ab}}$ is called an *admissible* finite function.

3.10. Lemma.
Let $X$, $Y$ be algebraic cpo's. If $Adm(\mathbf{a},\mathbf{b})$, then $f_{\mathbf{ab}}$ is welldefined, continuous and compact (in $[X \to Y]$).
Proof.i) Assume $Adm(\mathbf{a},\mathbf{b})$, $x\in X$. Now $Z_x=\{b_i \mid a_i\leq x\}$ is directed (for if $b_i,b_j\in Z_x$, then $a_i,a_j\leq x$, so $a_i,a_j\leq a_k\leq x$ for some $k$, and $b_i,b_j\leq b_k\in Z_x$ ), so $f_{\mathbf{ab}}$ is welldefined. Also $f_{\mathbf{ab}}(\bigvee Z) = \bigvee\{b_i \mid a_i\leq\bigvee Z\} = \bigvee\{b_i \mid \exists z\in Z(a_i\leq z)\} = \bigvee\{\bigvee\{b_i \mid a_i\leq z\} \mid z\in Z\} = \bigvee f_{\mathbf{ab}}(Z)$ for directed $Z$, so $f_{\mathbf{ab}}$ is continuous. Compactness remains to be shown, so assume $G\subseteq[X\to Y]$ directed, $f_{\mathbf{ab}}\leq\bigvee G$. This means $\forall x\in X\exists g\in G(f_{\mathbf{ab}}(x) \leq gx)$, i.e. $\forall x\in X\exists g\in G(\bigvee\{b_i \mid a_i\leq x\} \leq gx)$, hence $\forall j\leq|\mathbf{a}|\exists g\in G(\bigvee\{b_i \mid a_i\leq a_j\} \leq ga_j)$, so ($G$ is directed) $\exists g\in G\forall j\leq|\mathbf{a}|(\bigvee\{b_i \mid a_i\leq a_j\} \leq g(a_j))$, i.e. $\forall j\leq|\mathbf{a}|(f_{\mathbf{ab}}(a_j)\leq g(a_j))$ for some $g\in G$, which implies (by the definition of $f_{\mathbf{ab}}$) $f_{\mathbf{ab}}\leq g$; we conclude that $f_{\mathbf{ab}}$ is compact. []

3.11. Counterexample.
It is tempting to think that *any* function with a finite range of compact elements is compact, but this

is not the case: we give a counterexample, using the cpo's $N$ and $\underline{2}$ defined in 4.1.
Let $f, f_0, f_1, \ldots : N \to \underline{2}$ be defined by

$$f(x) = \bot \text{ if } x=\bot, \text{ otherwise } \top$$
$$f_n(x) = \top \text{ if } 0\leq x\leq n, \text{ otherwise } \bot$$

then $\{f_n \mid n\in\omega\}$ is directed and $f = \bigvee\{f_n \mid n\in\omega\}$, but there is no $f_n$ with $f \leq f_n$, so $f$ is not compact, although its range is a finite set of compact elements.

To prove that $[X \to Y]$ is algebraic (lemma 3.14), we need the supremum of subsets of $X$ and $Y$ which are, in general, not directed; they share, however, a property we define now:

3.12. Definition.
i) $B(X)$, the collection of *bounded* subsets of a cpo $X$, is defined by

$$B(X) := \{Y\subseteq X \mid \exists x\in X \forall y\in Y(y\leq x)\}.$$

ii) $\langle X,\leq\rangle$ is called *boundedly complete* if each $Y\in B(X)$ has a supremum, i.e. $\bigvee$ is defined for all bounded subsets of $X$.
iii) $\langle X,\leq\rangle$ is called a *domain* if it is an algebraic and boundedly complete cpo.
iv) If $\{x,y\}\in B(X)$ (i.e. $\exists z\in X(x\leq z \wedge y\leq z)$) then we write $x\uparrow y(X)$.
v) We write $x\vee y$ for $\bigvee\{x,y\}$.

3.13. Fact.
If $X$ is boundedly complete and $x\uparrow y(X)$ holds, then $x\vee y$ exists.

3.14. Lemma.
Let $X,Y$ be domains.
i) The compact elements of $[X \to Y]$ are exactly the admissible finite functions.
ii) $[X \to Y]$ is algebraic.
iii) $[X \to Y]$ is boundedly complete, hence a domain.
Proof. i) That admissible finite functions are compact was shown in lemma 3.10; we now prove the converse. Let $f\in[X\to Y]$, and define $G := \{g\in[X\to Y] \mid g \text{ admissible and finite, } g\leq f\}$.If

(1) $G$ is directed, and
(2) $f\leq\bigvee G$,

then, if $f$ is compact, $\exists g\in G(f\leq g)$, so $f=g$, and we are done.
Directedness of $G$ is obtained as follows. Let $g_1=f_{\mathbf{ab}}\in G$, $g_2=f_{\mathbf{cd}}\in G$, assuming, without loss of generality, that $a_1=c_1=\bot\in X$, $b_1=d_1=\bot\in Y$. Define $\mathbf{p},\mathbf{q}$ as the sequences satisfying

$\{<p_i,q_i> \mid i\leq|\mathbf{p}|\} = \{<a_i\vee c_j, b_i\vee d_j> \mid i\leq|\mathbf{a}|, j\leq|\mathbf{c}|, a_i\uparrow c_j(X))\}$. This is a correct definition, for if $a_i,c_j\leq x$ then $a_i\vee c_j$ exists (X is boundedly complete), and also $fa_i,fc_j\leq fx$, so $ga_i,hc_j\leq fx$ (for $g,h\leq f$), i.e. $b_i,d_j\leq fx$ and $b_i\vee d_j$ exists (Y is boundedly complete). Adm(**p**,**q**) follows easily from Adm(**a**,**b**) and Adm(**c**,**d**), so $h:=f_{\mathbf{pq}}$ is a admissible finite function. Also $g_1(x) = \bigvee\{b_i \mid a_i\leq x\} = \bigvee\{b_i\vee\perp \mid a_i\vee\perp\leq x\} \leq \bigvee\{b_i\vee d_j \mid a_i\vee c_j\leq x\} = h(x)$ and, analogously, $g_2(x)\leq h(x)$, so $g_1,g_2\leq h$. We conclude that G is directed, i.e. (1).
To see that $f\leq\bigvee G$ observe $fx = \bigvee\Downarrow_c(f(\bigvee\Downarrow_c(x))) = \bigvee\Downarrow_c(\bigvee f\Downarrow_c(x)) =$
$\bigvee\{b\in c(Y) \mid b\leq\bigvee\{fa \mid a\in c(X), a\leq x\}\} = \bigvee\{b\in c(Y) \mid \exists a\in c(X)(b\leq fa)\} =$
$\bigvee\{f_{ab}(x) \mid a\in c(X), b\in c(Y), b\leq fa\} \leq \bigvee\{gx \mid g\in G\}$, so we have (2).
ii) This follows from the proof of (i), using compact = (admissible and finite).
iii) Easy, using the pointwise definition of $\leq$ and $\bigvee$ in $[X\to Y]$. []

We now show that the restriction to domains is essential to obtain closure of algebraic cpo's under $\to$.

3.15. Lemma.
Let $<X,\leq>$ be an algebraic cpo. Then the following are equivalent:
i) X is a domain;
ii) $[X\to X]$ is an algebraic cpo;
iii) $\Downarrow_c(\lambda x.x) \subseteq c([X\to X])$ is directed;
iv) $\forall xy\in c(X)(x\uparrow y(X) \to x\vee y$ exists and $x\vee y\in c(X))$.
Proof. (i) $\Rightarrow$ (ii): Assume X is a domain, and let $f\in[X\to X]$. We have to show $\Downarrow_c(f)=\{g\in c([X\to X]) \mid g\leq f\}$ is directed, with supremum f. By lemma 3.14 we get $\Downarrow_c(f)=\{g$ admissible and finite $\mid g\leq f\}$, and we see that this set is directed and that its supremum equals f by inspection of the proof of (1) resp. (2) in the same lemma.
(ii) $\Rightarrow$ (iii): Evident, by the definition of algebraic cpo.
(iii) $\Rightarrow$ (iv): Assume $x,y\in c(X)$, $z\in X$, $x,y\leq z$. We have $f_{xx},f_{yy}\in c([X\to X])$ and $f_{xx},f_{yy}\leq\lambda x.x$, so $f_{xx},f_{yy}\in\Downarrow_c(\lambda x.x)$. This last set is directed, so we have a $f_{\mathbf{ab}}\in\Downarrow_c(\lambda x.x)$ with $f_{xx},f_{yy}\leq f_{\mathbf{ab}}$. Without loss of generality we assume $\mathbf{a}=\mathbf{b}$, $a_1=x$, $a_2=y$. Let $a\in\mathbf{a}$ satisfy $x,y\leq a$ and $\forall a'\in\mathbf{a}(x,y\leq a'\leq a \to a'=a)$: such an a always exists, for **a** finite and Adm(**a**,**a**). We claim: $x\vee y=a$. $x,y\leq a$ already holds; rests minimality, so assume $x,y\leq z\leq a$, then (by Adm(**a**,**a**)) $x,y\leq a''\leq z\leq a$ for some $a''\in\mathbf{a}$, hence (by the choice of a) $a''=a$ and $z=a$, and the claim is proved.
(iv) $\Rightarrow$ (i): If $Y\subseteq X$ is bounded, then so is $Z:=\{z\in c(X) \mid \exists y\in Y(z\leq y)\}$, hence every finite subset $F=\{z_1,...,z_n\}$ of Z has a supremum $z_1\vee...\vee z_n$ which is smaller than any upper bound of Y. Now $Z^*:=\{\bigvee F \mid F\subseteq Z, F$ finite$\}$ is welldefined and directed, and $\bigvee Z^*$ is both an upper bound of Y and smaller than any upper bound of Y, i.e. $\bigvee Z^*=\bigvee Y$. []

So far for the class of cpo's needed for CI, the concrete interpretation of POLYREC. AI, the abstract interpretation, requires one more restriction: $\bigvee Y$ has to be defined for *all* subsets of the cpo's involved. The reason for this has to do with the abstraction function abs between CI and

AI, and is explained in 1.2.

3.16. Definition.

i) Let $\langle X,\leq\rangle$ be a cpo. $\langle X,\leq\rangle$ is called a *complete lattice* if each $Y\subseteq X$ has a supremum.

ii) Let $\langle X,\leq\rangle$ be a complete lattice, $Y\subseteq X$. Then $\bigwedge Y$, *the least upper bound (infimum)* of Y, is defined by

$$\bigwedge Y := \bigvee\{x\in X \mid \forall y\in Y(x\leq y)\}.$$

iii) Let $\langle X,\leq\rangle$ be a cpo. If there is a $y\in X$ with $\forall x\in X(x\leq y)$, then we call this y the *top elemen* of X, denoted by $\top$.

iv) Let X, Y be complete lattices. We call $f : X \to Y$ *additive* if $f(\bigvee Z) = \bigvee(fZ)$ for all $Z\subseteq X$.

3.17. Lemma.

i) X is a domain with a top element $\Leftrightarrow$ X is an algebraic complete lattice.

ii) Let X, Y be algebraic complete lattices. Then the compact elements of $[X \to Y]$ are exactly the finite ones.

Proof. i) Easy, by the definition of boundedly complete.

ii) By lemma 3.14 it suffices to show that every finite function is equal to an admissible finite one. So let $\mathbf{a} = a_1,...,a_n$, $\mathbf{a} \subseteq X$, $\mathbf{b} = b_1,...,b_n$, $\mathbf{b} \subseteq Y$; put

$\mathbf{p}$ := the sequence of all disjunctions of subsets of $\mathbf{a}$ (so $\mathbf{p}$ has $2^n$ elements),
$\mathbf{q}$ is defined by $q_i := \bigvee\{b_j \mid b_j \leq p_i, j\leq n\}$ for $j = 1,...,2^n$,

then one easily verifies

$\mathbf{p}, \mathbf{q}$ contain only compact elements (by 3.15.(iv));
Adm($\mathbf{p}$,$\mathbf{q}$);
$\bigvee\{b_i \mid a_i \leq x\} = \bigvee\{q_i \mid p_i \leq x\}$ for all $x\in X$, i.e. $f_{\mathbf{ab}} = f_{\mathbf{pq}}$. []

To complete the picture of domains, we give a topological characterisation in terms of Ershov's $f_0$-spaces [Er75].

3.18. Definition.

Let $\langle X,\mathbf{O}(X)\rangle$ be a topological space.

i) The ordering $\leq$ on X, induced by $\mathbf{O}(X)$,is defined by

$$x\leq y := \forall O\in \mathbf{O}(X)(x\in O \to y\in O).$$

ii) The collection $X_0$ of *f-elements* of $X$ is defined by

$$X_0 := \{x \in X \mid \Uparrow(x) \in \mathbf{O}(X)\}$$

where $\Uparrow(x)$, the cone above x, is defined by $\Uparrow(x) := \{y \in X \mid x \leq y\}$.
iii) $\langle X,\mathbf{O}(X)\rangle$ is called an $f_0$*-space* if:
a) $X$ contains a least element $\bot$ w.r.t. $\leq$;
b) if two f-elements of X are consistent, their supremum exists and is again an f-element, i.e.

$$\forall x,y \in X_0(\exists z \in X(x \leq z \wedge y \leq z) \rightarrow x \vee y \text{ exists} \wedge x \vee y \in X_0),$$

where $x \vee y := \bigvee\{x,y\}$.
c) every open subset is the union of the cones above its f-elements, i.e.:

$$\forall O \in \mathbf{O}(X)(O = \cup\{\Uparrow(x) \mid x \in O \cap X_0\}).$$

3.19. Theorem.
Let $\langle X,\leq\rangle$ be a cpo and let $\mathbf{O}(X)$ be its topology. Then

$$\langle X,\mathbf{O}(X)\rangle \text{ is an } f_0\text{-space} \Leftrightarrow \langle X,\leq\rangle \text{ is a domain.}$$

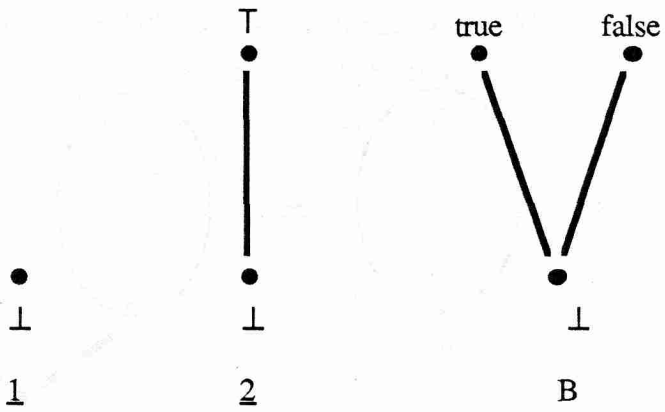
Proof. From $x \in X_0 \Leftrightarrow \{y \in X \mid x \leq y\} \in \mathbf{O}(X) \Leftrightarrow \forall Y \in D(X)(x \leq \bigvee Y \rightarrow \exists y \in Y(y \leq x)) \Leftrightarrow x \in c(X)$ it follows that the compact elements are exactly the f-elements. We use this in the rest of the proof. Now assume that $\langle X,\mathbf{O}(X)\rangle$ is an $f_0$-space. Then $\langle X,\leq\rangle$ is algebraic, for $\Downarrow_c(x) = \{y \in c(X) \mid y \leq x\}$ is closed under $\vee$ (by 3.18.(iii)(b)) and hence directed; also $\bigvee \Downarrow_c(x) = x$, for $\bigvee \Downarrow_c(x) \leq x$ by the definition of $\Downarrow_c(x)$, and if not$(x \leq \bigvee \Downarrow_c(x))$, then $x \in \{y \mid$ not$(y \leq \bigvee \Downarrow_c(x))\}$ (which is open, by 3.3), so (by 3.18.(iii)(c)) $\exists z \in X_0($not$(z \leq \bigvee \Downarrow_c(x)) \wedge z \leq x)$, which gives contradiction with $z \in \Downarrow_c(x)$. $\langle X,\leq\rangle$ is boundedly complete, too, for if $Y \subseteq X$ is bounded, then so is $Z := \{z \in X_0 \mid \exists y \in Y(z \leq y)\}$, hence every finite subset $F$ of $Z$ has a supremum which is smaller than any uppe bound of $Y$, so $Z^* := \{\bigvee F \mid F \subseteq Z, F \text{ finite}\}$ is welldefined and directed, and $\bigvee Z^*$ is both an upper bound of $Y$ and smaller than any upper bound of Y, i.e. $\bigvee Z^* = \bigvee Y$.
To prove the converse, let $\langle X,\leq\rangle$ be a domain. Then $x \vee y$ is defined for every bounded pair of compact elements, and if $Z$ is directed and $x \vee y \leq \bigvee Z$, then $x,y \leq \bigvee Z$, so there are $z, z' \in Z$ with $x \leq z$ and $y \leq z'$, hence ($Z$ is directed) $x,y \leq z''$ for some $z'' \in Z$, so $x \vee y \leq z''$ and we conclude that $x \vee y$ is compact, which settles the second clause of the definition of $f_0$-spaces. To obtain the third, we have to show $x \in O \in \mathbf{O}(X) \rightarrow \exists y \in O \cap X_0(y \leq x)$, which follows from $x = \bigvee \Downarrow_c(x)$ (for $X$ is algebraic), hence $\Downarrow_c(x) \cap O \neq \varnothing$ (by the definition of $\mathbf{O}(X)$). []

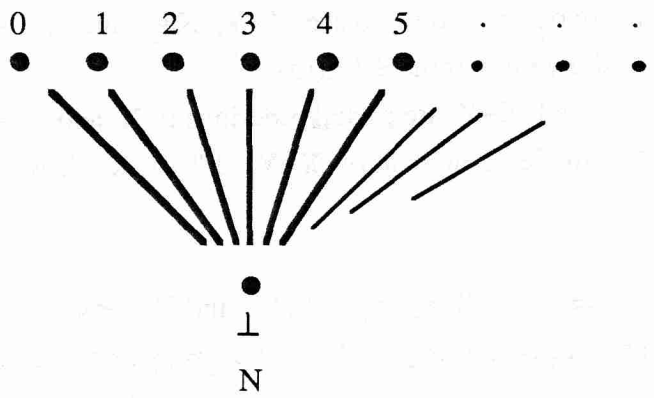
## § 4. OPERATIONS AND FUNCTIONS ON DOMAINS.

In this section we study operations and functions defined on domains and complete lattices, to be used for the interpretation of the type constructors $\times$, $+$, $\rightarrow$ and the (polymorphic) term constants of POLYREC. First we define some domains for the interpretation of N and B.

4.1. Definition. $\underline{1}$, $\underline{2}$, B and N are defined by

$\underline{1} := <\{\bot\},\leq>$ with $\bot\leq\bot$
$\underline{2} := <\{\bot,\top\},\leq>$, with $x\leq y$ iff $x=\bot$ or $y=\top$
$B := <\{\bot, \text{true},\text{false}\},\leq>$, with $x\leq y$ iff $x=\bot$ or $x=y$
$N := <\{\bot,0,1,2,3,...\},\leq>$, with $x\leq y$ iff $x=\bot$ or $x=y$




4.2. Lemma.
i) $\underline{1}$ and $\underline{2}$ are algebraic complete lattices.
ii) B and N are domains.
Proof. Easy. []

4.3. Definition.

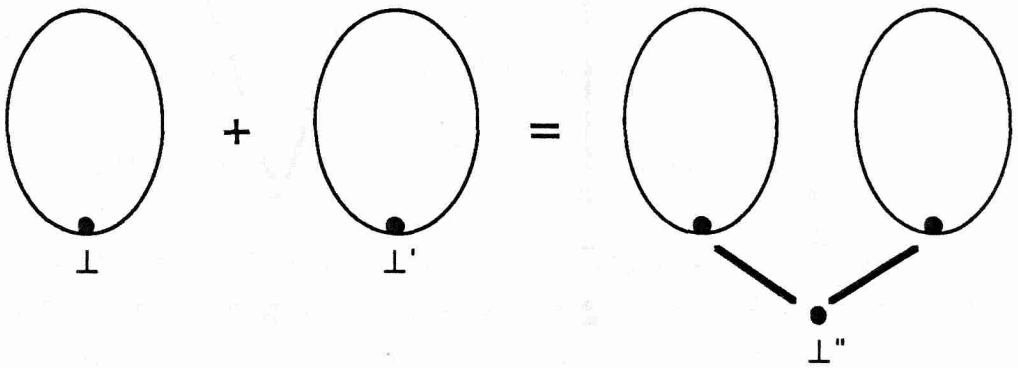
Let $\langle X,\leq\rangle$, $\langle X',\leq'\rangle$ be orderings.

The *product* $\langle X,\leq\rangle \times \langle X',\leq'\rangle = \langle X'',\leq''\rangle$ is defined by

$$X'' := X\times X'$$
$$\langle x,x'\rangle \leq'' \langle y,y'\rangle \text{ iff } x \leq y \text{ and } x' \leq' y'.$$

The *sum* $\langle X,\leq\rangle + \langle X',\leq'\rangle = \langle X'',\leq''\rangle$ is defined by

$$X+X' := \{\langle 0,x\rangle \mid x\in X\}\cup\{\langle 1,x'\rangle \mid x'\in X'\}\cup\{\bot''\}$$
$$a \leq'' b \text{ iff } a=\langle 0,x\rangle, b=\langle 0,y\rangle \text{ and } x\leq y$$
$$\text{or } a=\langle 1,x\rangle, b=\langle 1,y\rangle \text{ and } x\leq' y$$
$$\text{or } a=\bot''$$




4.4. Lemma. If $X$ and $Y$ are domains then so are $X\times Y$, $X+Y$ and $[X\to Y]$.

Proof. For $[X\to Y]$ this is shown in lemma 3.14.(iii).

One easily checks that $X\times Y$ and $X+Y$ are partial orderings if $X$ and $Y$ are. Likewise one observes that $\langle \bot_X,\bot_Y\rangle$ is the least element of $X\times Y$, $\bot''$ of $X+Y$. As to bounded sets and suprema, we have:

(1) $Z \in B(X\times Y) \Rightarrow Z_0 \in B(X), Z_1 \in B(Y)$ and $\bigvee Z=\langle \bigvee Z_0, \bigvee Z_1\rangle$,
where $Z_0:=\{x \mid \langle x,y\rangle \in Z\}$ and $Z_1:=\{y \mid \langle x,y\rangle \in Z\}$;

(2) $Z \in B(X+Y) \Rightarrow Z=\{\bot''\}$ and $\bigvee Z=\bot''$, or
$Z\subseteq\{\bot''\}\cup\{\langle 0,x\rangle \mid x\in X\}$, $Z_0 \in B(X)$ and $\bigvee Z = \langle 0, \bigvee Z_0\rangle$, or
$Z\subseteq\{\bot''\}\cup\{\langle 1,y\rangle \mid y\in Y\}$, $Z_1 \in B(X)$ and $\bigvee Z = \langle 1, \bigvee Z_1\rangle$,

where $Z_0 := \{x \mid \langle 0,x\rangle \in Z\}$, $Z_1 := \{y \mid \langle 1,y\rangle \in Z\}$; we conclude that $X\times Y$ and $X+Y$ are

boundedly complete cpo's.
To see that $X \times Y$ and $X+Y$ are algebraic, use

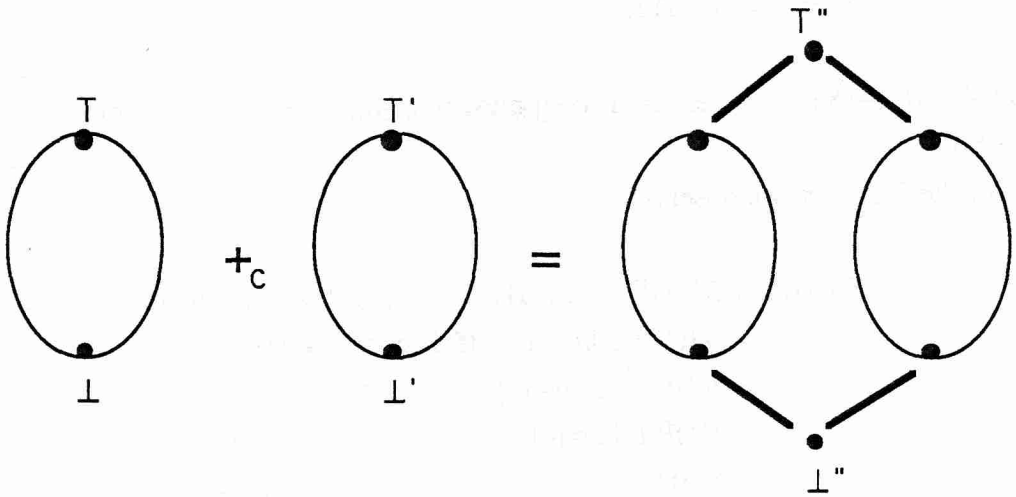
(3) $\langle x,y\rangle$ compact $\Leftrightarrow$ $x$ and $y$ compact.

(4) $a \in X+Y$ compact $\Leftrightarrow$ $a=\bot$ or $a=\langle i,x\rangle$ with $i \in \{0,1\}$ and $x$ compact. []

If $X$ and $Y$ are complete lattices, then $X+Y$ is in general not a complete lattice, for it fails to have a unique top element (unless $X$ or $Y$ is equal to $\underline{1}$). Therefore we define the complete sum of two orderings.

4.5. Definition.
Let $\langle X,\leq\rangle$, $\langle X', \leq'\rangle$ be orderings. The *complete sum* $\langle X,\leq\rangle +_c \langle X', \leq'\rangle = \langle X'',\leq''\rangle$ is defined by

$$X+_cX' := \{\langle 0,x\rangle \mid x \in X\} \cup \{\langle 1,x'\rangle \mid x' \in X'\} \cup \{\bot'', \top''\}$$
$$a \leq'' b \text{ iff } a=\langle 0,x\rangle, b=\langle 0,y\rangle \text{ and } x \leq y$$
$$\text{or } a=\langle 1,x\rangle, b=\langle 1,y\rangle \text{ and } x \leq' y$$
$$\text{or } a=\bot''$$
$$\text{or } b=\top''$$




4.6. Lemma. If $X$ and $Y$ are algebraic complete lattices then so are $X \times Y$, $X+_cY$ and $[X \to Y]$.
Proof. By lemma 3.17.(i) and lemma 4.4 it suffices to show that $X \times Y$ and $[X \to Y]$ have a top element: for $X \times Y$ this is $\langle \top_X, \top_Y\rangle$, and for $[X \to Y]$ this is $\lambda x.\top_Y$.
For $X+_cY$ we argue as follows. We have

$Z \subseteq X+_cY \Rightarrow Z \in B(X+Y)$ and $\bigvee Z$ is defined as in (2) in lemma 4.4, or
$\{x \mid <0,x> \in Z\} \neq \emptyset$, $\{y \mid <1,y> \in Z\} \neq \emptyset$ and $\bigvee Z = \top$",

so $X+_cY$ is a complete lattice, and with

$a \in X+_cY$ compact $\Leftrightarrow$ $a=\bot$" or $a=\top$" or $a=<i,x>$ ($i \in \{0,1\}$) and $x$ compact

one easily proves that $X+_cY$ is algebraic. []

4.7. Definition.
Let $X$ be a cpo. $fix : [X \to X] \to X$, the *fixpoint operator of* $X$, is defined by

$$fix(f) := \bigvee\{f^n\bot \mid n \in \omega\}$$

where $f^0x := x$, $f^{n+1}x := f(f^nx)$.

4.8. Lemma.
i) $\bigvee\{f^n\bot \mid n \in \omega\}$ is directed, so $fix(f)$ is welldefined.
ii) $fix$ is the *least* fixpoint operator, i.e.

$$f(fix(f)) = fix(f), \text{ and}$$
$$fx=x \Rightarrow fix(f) \leq x.$$

iii) $fix \in [[X \to X] \to X]$, i.e. $fix$ is continuous.
Proof. i) Easy.
ii) For the first part, we observe

$$\begin{aligned} f(fix(f)) &= f(\bigvee\{f^n\bot \mid n \in \omega\}) \\ &= \bigvee f(\{f^n\bot \mid n \in \omega\}) \quad (f \text{ is continuous}) \\ &= \bigvee\{f^{n+1}\bot \mid n \in \omega\} \\ &= \bigvee\{f^n\bot \mid n \in \omega\} \\ &= fix(f). \end{aligned}$$

For the second part, assume $fx=x$. We have $\bot \leq x$, and $y \leq x \Rightarrow fy \leq fx=x$ (for $f$ is continuous, hence monotonic); with induction this yields $\forall n \in \omega(f^n\bot \leq x)$. So $fix(f) = \bigvee\{f^n\bot \mid n \in \omega\} \leq x$.
iii) This follows from $fix(f) = \bigvee\{\lambda f.f^n\bot \mid n \in \omega\}$ and the fact that the $\lambda f.f^n\bot$ are continuous. []

4.9. Definition.

Let X, Y, Z be cpo's. We define the following functions:

$ite: B \to X \to X \to X$ (ite abbreviates **if then else**)

$ite(true,x,y) = x$

$ite(false,x,y) = y$

$ite(\bot,x,y) = \bot$

$pair: X \to Y \to X \times Y$

$pair(x)(y) := \langle x,y \rangle$

$p0: (X \times Y) \to X$

$p0(\langle x,y \rangle) := x$

$p1: (X \times Y) \to Y$

$p1(\langle x,y \rangle) := y$

$in0: X \to X+Y$

$in0(x) := \langle 0,x \rangle$

$in1: Y \to X+Y$

$in1(y) := \langle 1,y \rangle$

$case: [X \to Z] \to [Y \to Z] \to (X+Y) \to Z$

$case(f,g,\langle 0,x \rangle) := fx$

$case(f,g,\langle 1,y \rangle) := gy$

$case(f,g,\bot) := \bot$

4.10. Lemma.

ite, pair, p0, p1, in0, in1 and case are continuous.

Proof. Easy. □

4.11. Definition.

Let X, Y, Z be complete lattices. We define:

$ite_c: \underline{2} \to X \to X \to X$

$ite_c(\top,x,y) := x \vee y$

$ite_c(\bot,x,y) := \bot$

$in0_c: X \to (X +_c Y)$

$in0_c(x) := \langle 0,x \rangle$

$in1_c: Y \to (X +_c Y)$

$in1_c(y) := \langle 1,y \rangle$

$case_c$: $[X \to Z] \to [Y \to Z] \to [X +_c Y] \to Z$

$case_c(f,g,<0,x>) := fx$

$case_c(f,g,<1,y>) := gy$

$case(f,g,\bot) := \bot$

$case(f,g,\top) := \top$

4.12. Lemma.
$ite_c$, pair, p0, p1, $in0_c$, $in1_c$ and $case_c$ are additive.
Proof. Easy. []

4.13. Remark.
One might think that the *coalesced* sum +' , defined by

$$X +' Y := \{<0,x> \mid x \in X, x \neq \bot\} \cup \{<1,y> \mid y \in Y, y \neq \bot\} \cup \{\bot\},$$

together with

$in0'$: $X \to X+'Y$

$in0'(x) := <0,x>$ if $x \neq \bot$

$in0'(\bot) = \bot$

$in1'$: $Y \to X+'Y$

$in1'(y) := <1,y>$ if $y \neq \bot$

$in1'(\bot) = \bot$

could serve as an alternative for +, in0 and in1. This is the case within the context of this section, but not when interpreting POLYREC (more precisely: the polymorphic constant case). case is to be interpreted by case, and the axioms for case would require that

$$f\bot = case(f,g,in0'(\bot)) = case(f,g,\bot) = case(f,g,in1'(\bot)) = g\bot,$$

and this does not hold for all f and g.
A similar argument holds against the coalesced complete sum, defined by

$$X +'_c Y := \{<0,x> \mid x \in X, x \neq \bot, x \neq \top\} \cup \{<1,y> \mid y \in Y, y \neq \bot, y \neq \top\} \cup \{\bot,\top\}.$$

## § 5. CATEGORIES AND FUNCTORS.

In this section we introduce some categories, to be used for the interpretation of POLYREC. We start with a concise presentation of some basic notions of category theory.

### 5.1. Preliminaries.

A category $\mathbb{C}$ consists of a collection $Obj(\mathbb{C})$ of *objects* and a collection $Ar(\mathbb{C})$ of *arrows*. Furthermore we have:

the mappings $dom,cod : Ar(\mathbb{C}) \to Obj(\mathbb{C})$ (*domain* and *codomain*),
a partial binary operation $\circ$ (*composition*) on $Ar(\mathbb{C})$,
for every object $C \in \mathbb{C}$ an arrow $1_C \in Ar(\mathbb{C})$,

satisfying the following axioms:

$f \circ g$ is defined iff $dom(f)=cod(g)$;
$f \circ (g \circ h) = (f \circ g) \circ h$;
$dom(1_C) = cod(1_C) = C$;
$1_C \circ f = f$, $g \circ 1_C = g$.

We write $\mathbb{C}, \mathbb{C}', \ldots$ for categories, $C, C', \ldots$ for objects and $f,g,\ldots$ for arrows.

$f$ is called an *isomorphism* if there is a $g$ with $dom(g)=cod(f)$, $cod(g)=dom(f)$ and $f \circ g = 1_{cod(f)}$, $g \circ f = 1_{dom(f)}$. $C$ and $C'$ are called *isomorphic* (notation: $C \cong C'$) if there is an isomorphism $f$ with $dom(f)=C$, $cod(f)=C'$. Isomorphic objects in a category are identified.

It is possible to identify objects $C$ with the arrows $1_C$, and to consider $Obj(\mathbb{C})$ as a subset of $Ar(\mathbb{C})$. We shall do so sometimes.

$\mathbb{C}$ is a *subcategory* of $\mathbb{C}'$ (notation: $\mathbb{C} \subset \mathbb{C}'$) if $Obj(\mathbb{C}) \subset Obj(\mathbb{C}')$ and $Ar(\mathbb{C}) \subset Ar(\mathbb{C}')$. If, moreover

$$Ar(\mathbb{C}) = \{f \in Ar(\mathbb{C}') \mid dom(f), cod(f) \in Obj(\mathbb{C})\},$$

then $\mathbb{C}$ is a *full* subcategory of $\mathbb{C}'$.
$\mathbb{C}^{op}$, the *opposite* category of $\mathbb{C}$, is defined as the category satisfying:

$Obj(\mathbb{C}^{op}) = Obj(\mathbb{C})$;
there is a bijective mapping $op : \mathbb{C} \to \mathbb{C}^{op}$ with
$dom(op(f)) = cod(f)$, $cod(op(f)) = dom(f)$,
$op(f \circ g) = op(g) \circ op(f)$,
$op(1_C) = 1_C$.

Given two categories $\mathbb{C}$ and $\mathbb{C}'$, we can form the *product category* $\mathbb{C}*\mathbb{C}'$ as follows:

$Obj(\mathbb{C}*\mathbb{C}') = \{<C,C'> \mid C \in Obj(\mathbb{C}), C' \in Obj(\mathbb{C}')\}$
$Ar(\mathbb{C}*\mathbb{C}') = \{<f,g> \mid f \in Ar(\mathbb{C}), g \in Ar(\mathbb{C}')\}$
$dom(<f,g>) = <dom(f),dom(g)>$, $cod(<f,g>) = <cod(f),cod(g)>$
$<f,g> \circ <f',g'> = <f \circ f', g \circ g'>$
$1_{<C,C'>} = <1_C, 1_{C'}>$

This can be generalised to any number of categories. For $\mathbb{C}*...*\mathbb{C}$ (n times) we write $\mathbb{C}^n$.

A mapping F from $\mathbb{C}$ to $\mathbb{C}'$ is called a *functor* if F maps objects on objects and arrows on arrows, and commutes with the operations dom, cod and $\circ$, i.e.:

$F(dom(f)) = dom(F(f))$;
$F(cod(f)) = cod(F(f))$;
$F(f \circ g) = F(f) \circ F(g)$;
$F(1_C) = 1_{F(C)}$.

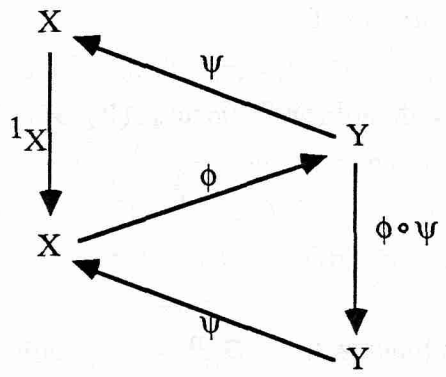
We write F, G, ... for functors. The collection of all functors from $\mathbb{C}$ to $\mathbb{C}'$ is denoted as $\mathbb{F}(\mathbb{C},\mathbb{C}')$. It is evident that the composition of two functors is again a functor.

Let $F, G \in \mathbb{F}(\mathbb{C},\mathbb{C}')$. A mapping $H : Obj(\mathbb{C}) \to Ar(\mathbb{C}')$ is called a *natural transformation between* F *and* G if, for all $f \in Ar(\mathbb{C})$:

$G(f) \circ H(dom(f)) = H(cod(f)) \circ F(f)$.

5.2. Definition.
Let X, Y be domains. The pair $<\phi,\psi>$ is called a *projection* of Y on X if:

$\phi \in [X \to Y]$,
$\psi \in [Y \to X]$,
$\psi \circ \phi = \lambda x.x$,
$\phi \circ \psi \leq \lambda y.y$.




We denote projections by $p = \langle\phi,\psi\rangle$, $p' = \langle\phi',\psi'\rangle$, $p'' = \langle\phi'',\psi''\rangle$, etc.

Now we define four categories.

5.3. Definition.

i) The category $\mathbb{D}$ is defined by:

Obj($\mathbb{D}$) := the collection of all *domains*;
Ar($\mathbb{D}$) := the collection of all *continuous functions* between elements of Obj($\mathbb{D}$);
$f \circ g := \lambda x \in dom(g).f(g(x))$, i.e. functional composition;
if $f \in Ar(\mathbb{D})$, $f : X \to Y$, then dom(f) := X, cod(f) := Y;
if $D \in Obj(\mathbb{D})$, then $1_D := \lambda x \in D.x$, the identity function on D.

ii) The category $\mathbb{A}$ is defined by:

Obj($\mathbb{A}$) := the collection of all *algebraic complete lattices*;
Ar($\mathbb{A}$) := the collection of all *additive functions* between elements of Obj($\mathbb{A}$);
$f \circ g := \lambda x \in dom(g).f(g(x))$, i.e. functional composition;
if $f \in Ar(\mathbb{A})$, $f : X \to Y$, then dom(f) := X, cod(f) := Y;
if $A \in Obj(\mathbb{A})$, then $1_A := \lambda x \in A.x$, the identity function on A.

iii) The categories $\mathbb{D}_p$ and $\mathbb{A}_p$ are defined by

$Obj(\mathbb{D}_p) = Obj(\mathbb{D})$, $Obj(\mathbb{A}_p) = Obj(\mathbb{A})$
$Ar(\mathbb{D}_p) = \{\langle\phi,\psi\rangle \mid \phi,\psi \in Ar(\mathbb{D}), \langle\phi,\psi\rangle \text{ is a projection}\}$
$Ar(\mathbb{A}_p) = \{\langle\phi,\psi\rangle \mid \phi,\psi \in Ar(\mathbb{A}), \langle\phi,\psi\rangle \text{ is a projection}\}$
$\langle\phi,\psi\rangle \circ \langle\phi',\psi'\rangle = \langle\phi\circ\phi',\psi'\circ\psi\rangle$
$dom(\langle\phi,\psi\rangle) = dom(\phi)$, $cod(\langle\phi,\psi\rangle) = cod(\phi)$
the unit arrows are of the form $\langle 1_D, 1_D\rangle$

iv) A functor $F \in \mathbb{F}(\mathbb{D}^n,\mathbb{D})$ is called *monotonic* if

if $f,g \in [X \to Y]$ with $f \leq g$, then $F(f) \leq F(g)$ (in $[F(X) \to F(Y)]$).

Analogously for $\mathbb{A}$ instead of $\mathbb{D}$.

5.4. Fact.
Every monotonic $F \in \mathbb{F}(\mathbb{D}^n,\mathbb{D})$ is also a functor from $\mathbb{D}_p{}^n$ to $\mathbb{D}_p$, with $F(\langle\phi_1,\psi_1\rangle,\ldots, \langle\phi_n,\psi_n\rangle) = \langle F(\phi_1,\ldots,\phi_n),F(\psi,\ldots,\psi_n)\rangle$. Idem for $\mathbb{A}$ instead of $\mathbb{D}$.

5.5. Definition.
i) We extend the operations $\times,+,\to$ on domains to mappings on categories of domains, and analogously for $+_c$ and algebraic complete lattices. Let $X,X',Y,Y'$ be domains, $f : X \to Y$, $g : X' \to Y'$. Then we define

$$f \times g : X \times X' \to Y \times Y'$$
$$(f \times g)\langle x,x'\rangle := \langle fx,gx'\rangle;$$

$$f + g : X + X' \to Y + Y'$$
$$(f+g)\langle 0,x\rangle := \langle 0,fx\rangle$$
$$(f+g)\langle 1,x'\rangle := \langle 1,gx'\rangle$$
$$(f+g)\bot = \bot;$$

$$f \to g : (Y \to X') \to (X \to Y')$$
$$(f \to g)x := g \circ x \circ f.$$

Assume moreover that $X,X',Y$ and $Y'$ are complete lattices. Then we define

$$f +_c g : X +_c X' \to Y +_c Y'$$
$$(f +_c g)\langle 0,x\rangle := \langle 0,fx\rangle$$
$$(f +_c g)\langle 1,x'\rangle := \langle 1,gx'\rangle$$
$$(f+g)\bot = \bot$$
$$(f+g)\top = \top.$$

ii) We extend $\times$, $+$ and $+_c$ to mappings on $\mathbb{D}_p{}^2$ resp. $\mathbb{A}_p{}^2$ as indicated in 5.4, and $\to$ as follows:

$$\langle\phi,\psi\rangle \to \langle\phi',\psi'\rangle := \langle\psi \to \phi',\phi \to \psi'\rangle$$

5.6. Lemma.

i) $\times \in \mathbb{F}(\mathbb{D}^2,\mathbb{D}), \mathbb{F}(\mathbb{A}^2,\mathbb{A}), \mathbb{F}(\mathbb{D}_p{}^2,\mathbb{D}_p), \mathbb{F}(\mathbb{A}_p{}^2,\mathbb{A}_p)$, and $\times$ is monotonic.

ii) $+ \in \mathbb{F}(\mathbb{D}^2,\mathbb{D}), \mathbb{F}(\mathbb{D}_p{}^2,\mathbb{D}_p)$, and $+$ is monotonic.

iii) $+_c \in \mathbb{F}(\mathbb{A}^2,\mathbb{A}), \mathbb{F}(\mathbb{A}_p{}^2,\mathbb{A}_p)$, and $+_c$ is monotonic.

iv) $\rightarrow \in \mathbb{F}(\mathbb{D}^{op}*\mathbb{D},\mathbb{D}), \mathbb{F}(\mathbb{A}^{op}*\mathbb{A},\mathbb{A}), \mathbb{F}(\mathbb{D}_p{}^2,\mathbb{D}_p), \mathbb{F}(\mathbb{A}_p{}^2,\mathbb{A}_p)$.

Proof. This follows from lemma 4.4 (Obj($\mathbb{D}$) closed under $\times$, $+$ and $\rightarrow$), lemma 4.6 (Obj($\mathbb{A}$) closed under $\times$, $+_c$ and $\rightarrow$), and:

a) if f and g are continuous, then so are $f\times g$, $f+g$ and $f\rightarrow g$;

b) if f and g are additive, then so are $f\times g$, $f+_c g$ and $f\rightarrow g$;

c) if p and p' are projections, then so are $p\times p'$, $p+p'$, $p+_c p'$ and $p\rightarrow p'$;

d) $\times,+,+_c$ and $\rightarrow$ commute with $\circ$;

e) $\times$, $+$ and $+_c$ are monotonic.

The proofs of (a) - (e) are simple exercises. []

## § 6. ABSTRACTIONS.

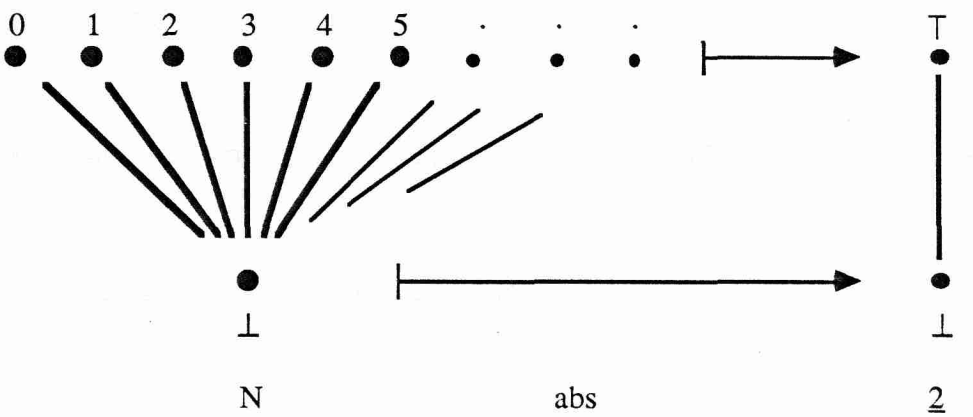
The categories $\mathbb{D}$ and $\mathbb{A}$, defined in the previous section, are to represent 'the real world' and 'the abstract world' of POLYREC, respectively, as we shall see in § 8. The relation between these worlds is given in terms of abstractions (arrows from objects in $\mathbb{D}$ to objects in $\mathbb{A}$) and abstraction transformations (a sort of natural transformations, see 6.10.(iii)). This section is devoted to these notions.

6.1. Definition.
Let $X, Y$ be domains, $f \in [X \to Y]$.
i) $f$ is called *strict* if $f\bot = \bot$, $\bot$-*reflecting* if $\forall x \in X(fx = \bot \Rightarrow x = \bot)$, and $\bot$-*unique* if it is both strict and $\bot$-reflecting.
ii) $f$ is called *compactness-preserving* if $fc$ is compact whenever $c$ is.
iii) $f$ is called an *abstraction* if $f$ is compactness-preserving, $\bot$-unique and $Y$ is an algebraic complete lattice.

6.2. Lemma.
i) $\lambda x.x \in [X \to X]$ is strict, and the composition of strict functions is strict. Idem for $\bot$-*reflecting*, $\bot$-*unique*, *compactness-preserving* or *being an abstraction* instead of *strict*.
ii) If $\langle \phi, \psi \rangle$ is an projection, then $\phi$ is $\bot$-unique and $\psi$ is strict.
Proof. i) Easy.
ii) $\phi\bot \leq \phi(\psi\bot) \leq \bot$, so $\phi\bot = \bot$; $\psi\bot = \psi(\phi\bot) = \bot$; if $\phi x = x$ then $x = \psi(\phi\bot) = \psi\bot = \bot$. []

6.3. Definition.
$abs_N$ and $abs_B$ are defined by

$$abs_N : N \to \underline{2}$$
$$abs_N(\bot) := \bot$$
$$abs_N(n) := \top \text{ if } n \neq \bot$$



$abs_B : B \to \underline{2}$

$abs_B(\bot) := \bot$

$abs_B(true) = abs_B(false) := \top$

6.4. Lemma.

$abs_N$ and $abs_B$ are abstractions.

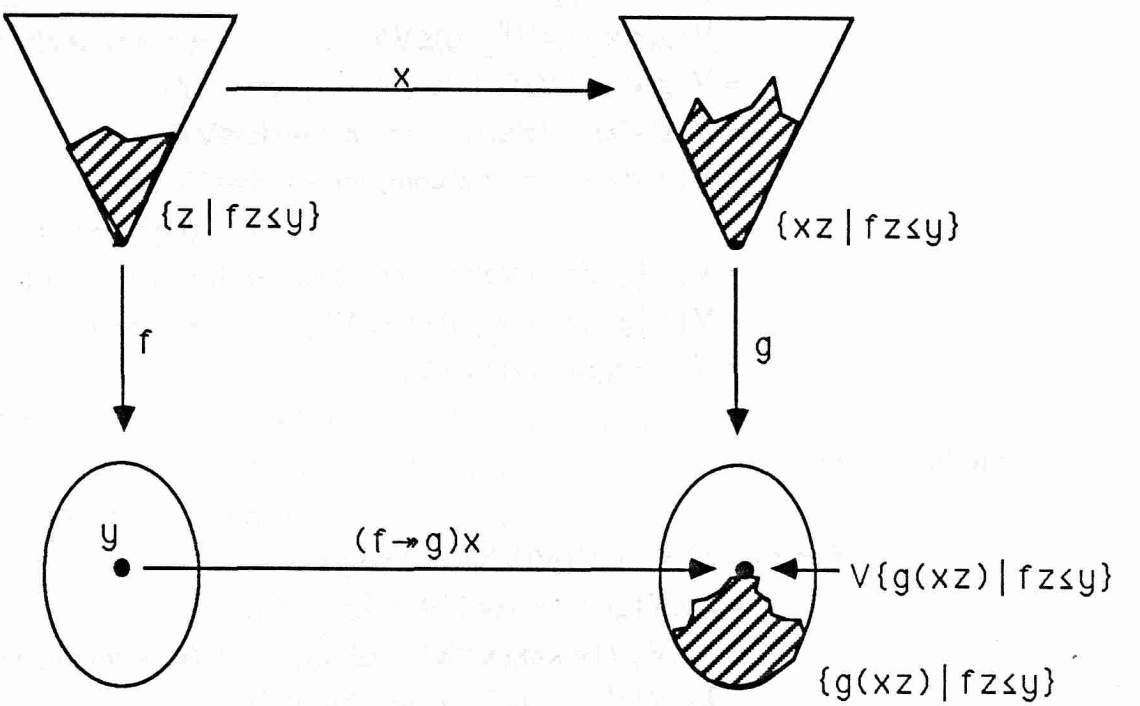
Proof. Easy. []

6.5. Definition.

i) We define two partial mappings $++$, $\twoheadrightarrow$ on $Ar(\mathbb{D})^2$ by

if $cod(f), cod(g) \in Obj(\mathbb{A})$, then $f++g$ is defined by $i \circ (f+g)$,
where $i$ is the embedding of $cod(f)+cod(g)$ into $cod(f)+_c cod(g)$,
$dom(f++g) = dom(f)+dom(g)$, $cod(f++g) = cod(f)+_c cod(g)$;

if $cod(g) \in Obj(\mathbb{A})$, then $f \twoheadrightarrow g$ is defined by $(f \twoheadrightarrow g)(x)y := \bigvee\{g(x(z)) \mid fz \leq y\}$
$dom(f \twoheadrightarrow g) = dom(f) \to dom(g)$, $cod(f \twoheadrightarrow g) = cod(f) \to cod(g)$.




ii) We call an n-ary (partial) mapping M from $Ar(\mathbb{D})^n$ to $Ar(\mathbb{D})$ *abstraction-preserving* if it is defined on all n-tuples of abstractions and maps these on abstractions.

6.6. Lemma.

i) $\times$, ++ and $\twoheadrightarrow$ are abstraction-preserving.

ii) The collection of abstraction-preserving mappings is closed under composition.

Proof. i) $\times$ : easy, using (3) in the proof of lemma 4.4 ($\langle x,y\rangle$ compact $\Leftrightarrow$ x and y compact) to show that $f\times g$ is compactness-preserving if f and g are.

++ : also quite easy, using (4) in the proof of lemma 4.4 (a compact $\Leftrightarrow$ $a=\bot$ or $a=\langle i,x\rangle$ with $i\in\{0,1\}$ and x compact).

$\twoheadrightarrow$ : this is shown in several steps. Assume that f,g are abstractions, $cod(g)\in Obj(\mathbb{A})$, $x\in[dom(f)\to dom(g)]$, $y\in cod(f)$. Now it suffices to show:

(1) $(f \twoheadrightarrow g)(x)(y)\in cod(g)$,

(2) $(f \twoheadrightarrow g)(x)\in [cod(f) \to cod(g)]$,

(3) $(f \twoheadrightarrow g)\in [[dom(f) \to dom(g)] \to [cod(f) \to cod(g)]]$,

(4) $(f \twoheadrightarrow g)$ preserves compactness,

(5) $(f \twoheadrightarrow g)$ is $\bot$- unique.

(1): easy (cod(g) is a complete lattice).

(2): follows from

$$
\begin{aligned}
(f \twoheadrightarrow g)(x)(\bigvee Y) &= \bigvee\{g(x(z)) \mid fz\le \bigvee Y\}\\
&= \bigvee\{g(x(z)) \mid f(\bigvee\Downarrow_c(z))\le \bigvee Y\} && \text{(cod(g) is algebraic)}\\
&= \bigvee\{g(x(z)) \mid \bigvee\{fz' \mid z'\le z,\ z' \text{ compact}\}\le \bigvee Y\}\\
&= \bigvee\{g(x(z)) \mid \forall z'\le z(z' \text{ compact} \to fz'\le \bigvee Y)\}\\
&= \bigvee\{g(x(z)) \mid \forall z'\le z(z' \text{ compact} \to \exists y\in Y(fz'\le y))\}\\
& && \text{(f preserves compactness)}\\
&= \bigvee\{\bigvee\{g(x(z)) \mid \forall z'\le z(z' \text{ compact} \to fz'\le y))\} \mid y\in Y\}\\
&= \bigvee\{\bigvee\{g(x(z)) \mid fz\le y))\} \mid y\in Y\}\\
&= \bigvee\{(f \twoheadrightarrow g)(x)(y) \mid y\in Y\}
\end{aligned}
$$

(3): follows from

$$
\begin{aligned}
(f \twoheadrightarrow g)(\bigvee X) &= \lambda y.\bigvee\{g((\bigvee X)(z)) \mid fz\le y\}\\
&= \lambda y.\bigvee\{g(\bigvee\{xz \mid x\in X\}) \mid fz\le y\}\\
&= \lambda y.\bigvee\{\bigvee\{g(xz) \mid x\in X\} \mid fz\le y\} && \text{(g is continuous)}\\
&= \lambda y.\bigvee\{\bigvee\{g(xz) \mid fz\le y\} \mid x\in X\}\\
&= \lambda y.\bigvee\{(f \twoheadrightarrow g)(x)(y) \mid x\in X\}\\
&= \bigvee\{(f \twoheadrightarrow g)(x) \mid x\in X\}
\end{aligned}
$$

(4): let x be compact, so $x = x_{\mathbf{ab}} = \lambda y.\bigvee\{b_i \mid a_i\le y\}$ with Adm(**a**,**b**), by lemma 3.14.(i). Then

$$
\begin{aligned}
(f \twoheadrightarrow g)(x) &= \lambda y.\bigvee\{g(xz) \mid fz \le y\} \\
&= \lambda y.\bigvee\{g(\bigvee\{b_i \mid a_i \le z\}) \mid fz \le y\} \\
&= \lambda y.\bigvee\{gb_i \mid a_i \le z,\ fz \le y\} \\
&= \lambda y.\bigvee\{gb_i \mid fa_i \le y\} \\
&= x_{\mathbf{fa},\mathbf{gb}}
\end{aligned}
$$

so $(f \twoheadrightarrow g)(x)$ is compact (by lemma 3.17.(ii), for $\mathbf{fa}$, $\mathbf{gb}$ are compact), and we see that $f \twoheadrightarrow g$ preserves compactness.
(5): follows from

$$
\begin{aligned}
(f \twoheadrightarrow g)(x) = \bot &\Leftrightarrow \forall y\ \bigvee\{g(xz) \mid fz \le y\} = \bot \\
&\Leftrightarrow \forall z\ (g(xz) = \bot) \\
&\Leftrightarrow \forall z\ (xz = \bot) \qquad\qquad (g \text{ is } \bot\text{-unique}) \\
&\Leftrightarrow x = \bot.
\end{aligned}
$$

ii) Straightforward. []

We now prove two important properties of $\twoheadrightarrow$, to be used in 8.13.

6.7. Lemma.
i) Let $f,g \in Ar(\mathbb{D})$, $cod(g) \in Obj(\mathbb{A})$, $x \in [dom(f) \to dom(g)]$, $y \in cod(f)$. Then

$$g(xy) \le ((f \twoheadrightarrow g)x)(fy).$$

ii) Let $f$ be an abstraction. Then

$$((f \twoheadrightarrow f) \twoheadrightarrow f)(fix_{dom(f)}) \le fix_{cod(f)}.$$

Remark. See 10.1.3 for a strengthening of this ineqality to an equality.
Proof. i) Follows from $g(xy) \le \bigvee\{g(xz) \mid fz \le fy\} = ((f \twoheadrightarrow g)x)(fy)$.
ii) Let $X := dom(f)$, $Y := cod(f)$, $a \in [Y \to Y]$. Then

$$
\begin{aligned}
((f \twoheadrightarrow f) \twoheadrightarrow f)(fix)a &= \bigvee\{f(fix(d)) \mid (f \twoheadrightarrow f)d \le a\} \\
&= \bigvee\{f(\bigvee\{d^n\bot \mid n \in \omega\}) \mid (f \twoheadrightarrow f)d \le a\} \\
&= \bigvee\{\bigvee\{f(d^n\bot) \mid n \in \omega\} \mid (f \twoheadrightarrow f)d \le a\} \\
&= \bigvee\{f(d^n\bot) \mid n \in \omega,\ (f \twoheadrightarrow f)d \le a\} \\
&= \bigvee\{f(d^n\bot) \mid n \in \omega,\ \forall y \in Y(\bigvee\{f(dx) \mid fx \le y\} \le ay)\} \\
&= \bigvee\{f(d^n\bot) \mid n \in \omega,\ \forall x \in X \forall y \in Y(fx \le y \to f(dx) \le ay)\} \\
&= \bigvee\{f(d^n\bot) \mid n \in \omega,\ d \in D\}, \text{ where}
\end{aligned}
$$

$$D := \{d \mid \forall x \in X \forall y \in Y(fx \leq y \rightarrow f(dx) \leq ay)\};$$

now, if $d \in D$, then $f(d^n \perp) \leq a^n \perp$ (proved with induction over n, using $f\perp = \perp$), so we have

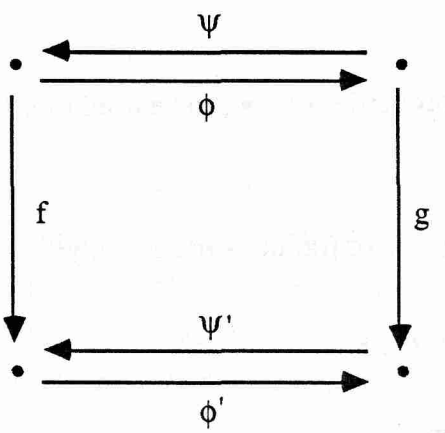
$$((f \twoheadrightarrow f) \twoheadrightarrow f)(fix)a \leq \bigvee\{a^n \perp \mid n \in \omega\} = fix(a). \qquad \square$$

We go one level higher and introduce abstraction transformations.

6.8. Definition.

i) Let $p=\langle\phi,\psi\rangle \in Ar(\mathbb{D}_p)$, $p'=\langle\phi',\psi'\rangle \in Ar(\mathbb{A}_p)$, and let $f,g \in Ar(\mathbb{D})$. We call $(p,p',f,g)$ an *abstraction diagram* if

$dom(p) = dom(f)$, $cod(p) = dom(g)$, $dom(p') = cod(f)$, $cod(p') = cod(g)$,
f and g are abstractions,
$\phi' \circ f = g \circ \phi$.




ii) Let $F \in \mathbb{F}(\mathbb{D}_p{}^n, \mathbb{D}_p)$, $G \in \mathbb{F}(\mathbb{A}_p{}^n, \mathbb{A}_p)$. Then the partial mapping $H : \mathbb{D}^n \rightarrow \mathbb{D}$ is called an *abstraction transformation between* F *and* G if the following holds:

if $(p_1,p'_1,f_1,g_1),\ldots, (p_n,p'_n,f_n,g_n)$ are abstraction diagrams, then so is $(F(\mathbf{p}),G(\mathbf{p'}),H(\mathbf{f}),H(\mathbf{g}))$.

Notation: $H : F \Rightarrow G$.

iii) $\mathbb{AT}$ ($\mathbb{AT}^n$) is the collection of (n-ary) abstraction transformations.

6.9. Remark.

The choice of $\phi' \circ f = g \circ \phi$ as commutation condition in the definition of abstraction diagram is dicussed in the Appendix.

6.10. Lemma.

i) If $(p,p',f,g)$ is an abstraction diagram, then $f\circ\psi \leq \psi'\circ g$.

ii) Abstraction transformations preserve abstractions.

iii) Abstraction transformations are natural transformations.

Proof. i) $f\circ\psi = \psi'\circ\phi'\circ f\circ\psi = \psi'\circ g\circ\phi\circ\psi \leq \psi'\circ g$.

ii) Use that $\langle\langle\lambda x\in \mathrm{dom}(f).x, \lambda x\in \mathrm{dom}(f).x\rangle, \langle\lambda x\in \mathrm{cod}(f).x, \lambda x\in \mathrm{cod}(f).x\rangle,f,f\rangle$ is an abstraction diagram whenever $f$ is an abstraction.

iii) We shall show (considering only the unary case): if $H : F \Rightarrow G$ and $G$ restricted to $\mathbb{A}_p$ is a functor on $\mathbb{A}_p$, then $H$ is a natural transformation from $F' \in \mathbb{F}(\mathbb{A}',\mathbb{A}')$ to $G' \in \mathbb{F}(\mathbb{A}',\mathbb{A}')$, where

$\mathbb{A}'$ is defined by $\mathrm{Obj}(\mathbb{A}') = \mathrm{Obj}(\mathbb{A}')$, $\mathrm{Ar}(\mathbb{A}') = \{\phi \mid \langle\phi,\psi\rangle \in \mathrm{Ar}(\mathbb{A})\}$,
$F'$ resp. $G'$ is the straightforward restriction of $F$ resp. $G$ to $\mathbb{A}'$.

Now, if $\phi \in \mathrm{Ar}(\mathbb{A}')$, then $p = \langle\phi,\psi\rangle \in \mathrm{Ar}(\mathbb{A})$ for some $\psi$, so $(p,p,1_{\mathrm{dom}(\phi)},1_{\mathrm{cod}(\phi)})$ is an abstraction diagram, and idem for $(Fp,Gp,H(\mathrm{dom}(\phi)),H(\mathrm{cod}(\phi)))$; so

$$G'(\phi)\circ H(\mathrm{dom}(\phi)) = H(\mathrm{cod}(\phi))\circ F'(\phi),$$

i.e. $H$ is a natural transformation from $F$ to $G$. □

6.11. Lemma.

i) $\times : \times \Rightarrow \times$.

ii) $++ : + \Rightarrow +_c$.

iii) $\twoheadrightarrow : \rightarrow \Rightarrow \rightarrow$.

iv) If $H : F \Rightarrow G$, $H' : F' \Rightarrow G'$ then $H\circ H' : F\circ F' \Rightarrow G\circ G'$, i.e. $\mathbb{AT}$ is closed under composition.

Proof. i), ii), iii): Let $(\langle\phi,\psi\rangle,\langle\phi',\psi'\rangle,f,g)$ and $(\langle\phi'',\psi''\rangle,\langle\phi''',\psi'''\rangle,f',g')$ be two abstraction diagrams, so they satisfy

$$\phi''\circ f = g\circ\phi \text{ and } \phi'''\circ f' = g'\circ\phi'.$$

We must show that

$$(\langle\phi\times\phi'',\psi\times\psi''\rangle,\langle\phi'\times\phi''',\psi'\times\psi'''\rangle,f\times f',g\times g'),$$
$$(\langle\phi+\phi'',\psi+\psi''\rangle,\langle\phi'+_c\phi''',\psi'+_c\psi'''\rangle,f++f',g++g'),$$
$$(\langle\phi\rightarrow\phi'',\psi\rightarrow\psi''\rangle,\langle\phi'\rightarrow\phi''',\psi'\rightarrow\psi'''\rangle,f\twoheadrightarrow f',g\twoheadrightarrow g')$$

are abstraction diagrams. By 6.6.(i), $f\times f'$, $g\times g'$, $f++f'$, $g++g'$, $f\twoheadrightarrow f'$ and $g\twoheadrightarrow g'$ are abstractions,

so it remains to see that

(6) $(\phi''\times\phi''')\circ(f\times f') = (g\times g')\circ(\phi\times\phi')$,

(7) $(\phi''+_C\phi''')\circ(f{+}{+}f') = (g{+}{+}g')\circ(\phi{+}\phi')$,

(8) $(\psi''\to\phi''')\circ(f\twoheadrightarrow f') = (g\twoheadrightarrow g')\circ(\psi\to\phi')$.

Now (6) and (7) are easily verified, and for (8) we argue as follows. We have

$$((\psi''\to\phi''')\circ(f\twoheadrightarrow f'))(x)y = \phi'''(\bigvee\{f'(xz) \mid fz\leq\psi''y\}),$$
$$((g\twoheadrightarrow g')\circ(\psi\to\phi'))(x)y = \bigvee\{g'(\phi'(x(\psi z))) \mid gz\leq y\} = \phi'''(\bigvee\{f'(x(\psi z)) \mid gz\leq y\}),$$

where the last equality holds because of $g'\circ\phi' = \phi'''\circ f'$ and the additivity of $\phi'''$, so it suffices to show

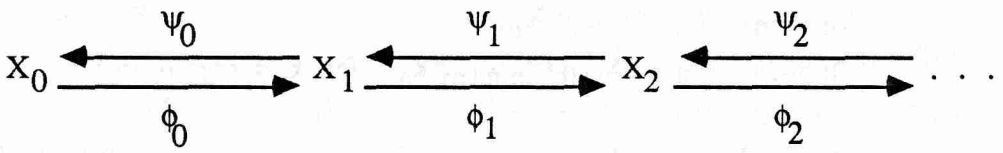
(9) $\{z \mid fz\leq\psi''y\} = \{\psi z \mid gz\leq y\}$,

which is done in two steps. First assume $z' \in \{z \mid fz\leq\psi''y\}$, i.e. $fz'\leq\psi''y$, then $\phi''(fz') \leq \phi''(\psi''y) \leq y$, so (by $g\circ\phi = \phi''\circ f$) $g(\phi z') \leq y$ and $z' = \psi(\phi z') \in \{\psi z \mid gz\leq y\}$. For the other way round, assume $u \in \{\psi z \mid gz\leq y\}$, then $u = \psi z$ for some $z$ with $gz\leq y$; then $\psi''(gz) \leq \psi''y$ and also (by lemma 6.10.(i)) $f\circ\psi \leq \psi''\circ g$, so $f(\psi z) \leq \psi''y$, so $u = \psi z \in \{z \mid fz\leq\psi''y\}$.
So (9) is proved.
iv) Straightforward. □

## § 7. CHAINS AND LIMITS.

For the definition of the interpretation of recursively defined types, we need a fixpoint construction defined on functors. This construction is based on limits of chains of cpo's. We start this section with the definition of the notions of chain and limit, and derive some properties. Then we define continuous functors as those functors which commute with taking limits of chains, and show how fixpoints for such functors can be obtained by a construction closely related to the definition of fixpoints of continuous functions on cpo's (see 4.7; the method is analogous to the one described in [SP82]). Finally we consider fixpoints of abstraction transformations.

7.1. Definition.
Let $<X_n>_n$ be a sequence of cpo's, and let, for all $n<\omega$, $p_n = <\phi_n,\psi_n>$ be a projection of $X_{n+1}$ on $X_n$. Then we call $<X_n,p_n>_n$ (also written $<X_n>_n$) a *chain*.




$\lim_n X_n = X$, the *limit* of $<X_n>_n$, is defined by

$$X := \{\mathbf{x}=<x_0,x_1,...> \mid \forall n\in\omega(\, x_n\in X_n \wedge \psi_n(x_{n+1})=x_n\,)\,\},$$
$$\mathbf{x}\leq\mathbf{y} \text{ iff } \forall n\in\omega(\, x_n \leq_n y_n\,).$$

We also put $Y_n := \{x_n \mid x \in Y\}\subseteq X_n$ for $Y\subseteq X$.
It is clear that chains are preserved under functors, i.e. if $<X^1{}_m>_m = <X^1{}_m,p^1{}_m>_m,...,$ $<X^n{}_m>_m = <X^n{}_m,p^n{}_m>_m$ are chains and $F\in \mathbb{F}(\mathbb{D}_p{}^n,\mathbb{D}_p)$ then

$$F(<X^1{}_m>_m,...,<X^n{}_m>_m) = <F(X^1{}_m,...,X^n{}_m),F(p^1{}_m,...,p^n{}_m)>_m$$

is a chain.

7.2. Lemma.
i) The limit of a chain of cpo's is a cpo.
ii) $\lim_n X_n = \lim_n X_{n+m}$.
Proof. i) It is evident that $\leq$ defined in 7.1 is a partial order. Furthermore $\bot := <\bot_0,\bot_1,...>$ is the smallest element; the supremum of $Y\in D(X)$ is defined by $\bigvee Y := <\bigvee Y_0,\bigvee Y_1,...>$, and one easily checks that $Y_n$ is directed whenever $Y$ is.

ii) This follows with induction from $\lim_n X_n \cong \lim_n X_{n+1}$. This isomorphism is realised by $i : \lim_n X_n \to \lim_n X_{n+1}$ and $j : \lim_n X_{n+1} \to \lim_n X_n$, defined by $i(\langle x_n\rangle_n) = \langle x_{n+1}\rangle_n$ and $j(\langle x_n\rangle_n) = \langle \psi_0(x_0)\rangle^*\langle x_n\rangle_n$. []

To prove stronger closure properties on limits of chains, we need some functions.

7.3. Definition.
Let $\langle X_n\rangle_n$ be a chain, $m,n\in\omega$. We define

$$\phi_{n,n+m}: X_n \to X_{n+m},$$
$$\psi_{n+m,n}: X_{n+m} \to X_n$$

inductively by

$$\phi_{n,n}(x_n) = x_n \text{ for } x_n\in X_n,$$
$$\phi_{n,n+m+1}(x_n) = \phi_{n+m}(\phi_{n,n+m}(x_n)) \text{ for } x_n\in X_n;$$

$$\psi_{n,n}(x_n) = x_n \text{ for } x_n\in X_n,$$
$$\psi_{n+m+1,n}(x_{n+m+1}) = \psi_{n+m,n}(\psi_{n+m}(x_{n+m+1})) \text{ for } x_{n+m+1}\in X_{n+m+1}.$$

For $x_n\in X_n$, the *extension* $\mathbf{e}(x_n)$ of $x_n$ is defined by

$$\mathbf{e}(x_n) := \langle e_0(x_n), e_1(x_n),\ldots\rangle, \text{ where}$$
$$e_n(x_n) := x_n,$$
$$e_m(x_n) := \psi_{n,m}(x_n) \text{ if } 0\le m<n,$$
$$e_m(x_n) := \phi_{n,m}(x_n) \text{ if } n<m<\omega.$$

7.4. Lemma.
Let $\langle X_n\rangle_n$ be a chain, $x_n\in X_n$.
i) $\mathbf{e}(x_n)\in \lim_n X_n$.
ii) Let $x,y\in X_n$, $x\le y$. Then $\mathbf{e}(x)\le\mathbf{e}(y)$.
iii) Let $\mathbf{x}=\langle x_n\rangle_n\in \lim_n X_n$. Then $\mathbf{e}(x_n) \le \mathbf{x}$ for every $n$.
iv) If $x_n$ is compact, then so is $\phi_n(x_n)\in X_{n+1}$.
v) If $x_n$ is compact, then so is $\mathbf{e}(x_n)$.
vi) If the $X_n$ are algebraic, then:

$$\langle x_n\rangle_n\in \lim_n X_n \text{ compact} \Leftrightarrow \exists n\ (x_n \text{ compact and } \langle x_n\rangle_n=\mathbf{e}(x_n)).$$

Proof. i),ii), iii): easy, using the properties of $\psi_n$ and $\phi_n$.

iv) $\phi_n(x_n) \leq \bigvee Y \Rightarrow x_n = \psi_n(\phi_n(x_n)) \leq \bigvee(\psi_n[Y]) \Rightarrow \exists y \in Y(x_n \leq \psi_n(y)) \Rightarrow$
$\exists y \in Y(\phi_n(x_n) \leq \phi_n(\psi_n(y)) \leq y)$, using the properties of $\psi_n$ and $\phi_n$.
v) If $e(x_n) \leq \bigvee Y$, then $x_n \leq \bigvee Y_n$, so $x_n \leq y_n$ for some $y \in Y$; but then (by (iii)) $x_n \leq y$, and from this the result follows.
vi) The $\Leftarrow$ part follows from (iv), so we prove the other part. Let $\langle x_n \rangle_n \in \lim_n X_n$ and consider

$$Z := \{e(z_n) \mid z_n \in X_n,\ z_n \text{ compact},\ z_n \leq x_n, n \in \omega\};$$

we claim

(1) $Z \leq \langle x_n \rangle_n$,
(2) $Z$ directed,
(3) $\bigvee Z = \langle x_n \rangle_n$.

(1) follows from (iv).
(2): let $e(z_m), e(z'_{m+n}) \in Z$, then $\phi_{m,m+n}(z_m), z'_{m+n} \in X_{m+n}$, compact (using (iv)) and $\leq x_{m+n}$ (for $\phi_{m,m+n}(z_m) \leq \phi_{m,m+n}(x_m) = \phi_{m,m+n}(\psi_{m+n,m}(x_{m+n})) \leq x_{m+n}$); so ($X_{m+n}$ is algebraic) there is a compact $z''_{m+n} \in X_{m+n}$ with $\phi_{m,m+n}(z_m), z'_{m+n} \leq z''_{m+n} \leq x_{m+n}$, hence $e(z_m) = e(\phi_{m,m+n}(z_m)) \leq e(z''_{m+n})$, $e(z'_{m+n}) \leq e(z''_{m+n})$ and $e(z''_{m+n})$ is compact (by (v)), so $e(z''_{m+n}) \in Z$.
(3) follows from $\bigvee Z_n = x_n$ for all $n \in \omega$, which is a consequence of $\{z \in X_n \mid z \leq x_n, z \text{ compact}\} \subseteq Z_n$ and the fact that $X_n$ is algebraic.
Now if $\langle x_n \rangle_n$ is compact then $\langle x_n \rangle_n \in Z$ by (3) and we are done. []

7.5. Theorem.
$\mathbb{D}$ and $\mathbb{A}$ are closed under taking limits of chains.
Proof. This is shown by proving the following for a chain $\langle X_n \rangle_n$:

(4) if the $X_n$ are algebraic, then so is $\lim_n X_n$;
(5) if the $X_n$ are boundedly complete, then so is $\lim_n X_n$;
(6) if the $X_n$ are complete lattices, then so is $\lim_n X_n$.

We prove (4) - (6).
(4) This follows directly from 7.4.(vi) and (1) - (3) in the proof of 7.4.(vi).
(5) Assume that the $X_n$ are boundedly complete, and let $Y \subseteq \lim_n X_n$ with $Y \leq x \in \lim_n X_n$. Then, for every $n < \omega$, $Y_n \leq x_n$, so $\bigvee Y_n$ exists; hence $\bigvee Y$ exists. So $\lim_n X_n$ is boundedly complete.
(6) Easy, by the fact that $\bigvee Y := \langle \bigvee Y_0, \bigvee Y_1, \ldots \rangle$. []

We want to obtain fixed points of functors. The method used here, inspired on the fixpoint construction in a cpo (see 4.7), is straightforward: start with the simplest cpo $\underline{1}$, construct the

sequence $<\underline{1},F(\underline{1}),F(F(\underline{1})),...>$, show that it is a chain and take the limit $\lim_n F^n(1)$. To see that this limit is indeed a fixpoint of F (i.e. $F(\lim_n F^n(\underline{1})) = \lim_n F^n(\underline{1})$), it suffices to know that $F(\lim_n X_n) = \lim_n F(X_n)$, i.e. applying F commutes with taking limits. Such functors are defined *continuous* in the next definition.

7.6. Definition.
Let $F \in \mathbb{F}(\mathbb{D}_p^n, \mathbb{D}_p)$. F is called *continuous* if for all chains $<X^1{}_n>_n,...,<X^n{}_n>_n$:

$$\lim_n F(X^1{}_n,...,X^n{}_n) = F(\lim_n X^1{}_n,...,\lim_n X^n{}_n).$$

Notation: $F \in \mathbb{CF}(\mathbb{D}_p^n, \mathbb{D}_p)$. Idem for $\mathbb{A}_p$ instead of $\mathbb{D}_p$.

7.7. Lemma.
i) $\times$, + and $\rightarrow$ are continuous.
ii) Continuous functors are closed under composition.
Proof. i) The continuity of $\times$ and + is easy; $\rightarrow$ requires more work. Let $<X_n>_n$, $<Y_n>_n$ be chains with $X:=\lim_n X_n$, $Y:=\lim_n Y_n$, $Z:= \lim_n [X_n \rightarrow Y_n]$. We shall give i,j with

(7) $\quad i : [[X \rightarrow Y] \rightarrow Z]$
(8) $\quad j : [Z \rightarrow [X \rightarrow Y]]$
(9) $\quad i \circ j = 1_Z$
(10) $\quad j \circ i = 1_{[X \rightarrow Y]}$;

then it follows that $\lim_n X_n \rightarrow \lim_n Y_n \cong \lim_n [X_n \rightarrow Y_n]$.
We recall:

$$X = \{\mathbf{x}=<x_n>_n \mid \forall n(x_n \in X_n \wedge \psi_n(x_{n+1})=x_n)\},$$
$$Y = \{\mathbf{y}=<y_n>_n \mid \forall n(y_n \in Y_n \wedge \psi'_n(y_{n+1})=y_n)\},$$
$$Z = \{\mathbf{g}=<g_n>_n \mid \forall n(g_n \in [X_n \rightarrow Y_n] \wedge \psi'_n \circ g_{n+1} \circ \phi_n = g_n)\}.$$

Now define i,j by

$$i(f) := <\lambda x_n.(f(e(x_n)))_n>_n \text{ for } f \in [X \rightarrow Y]$$
$$j(\mathbf{g}) := \lambda x.<\bigvee\{\psi'_{n+m,n}(g_{n+m}(x_{n+m})) \mid m \in \omega\}>_n \text{ for } \mathbf{g}=<g_n>_n \in Z.$$

We show (7) - (10).
(7): Let $f \in [X \rightarrow Y]$. We have to show that $<\lambda x_n.(f(e_n(x_n)))_n>_n \in Z$, i.e. for all n

$$\lambda x_n.(f(e(x_n)))_n \in [X_n \rightarrow Y_n] \text{ and}$$
$$\forall x_n \in X_n(\psi'_n(f(e(\phi_n(x_n))))_{n+1}=(f(e(x_n)))_n).$$

The first part is evident, the second is seen as follows: by $f \in [X \to Y]$ we have $\psi'_n(f(e(\phi_n(x_n))))_{n+1} = (f(e(\phi_n(x_n))))_n$ which is equal to $(f(e(x_n)))_n$, by the definition of **e**. The continuity of i is easily verified.

(8): Let $\mathbf{g}=\langle g_n\rangle_n \in Z$, $\mathbf{x}=\langle x_n\rangle_n \in X$. We have to show

$j(\mathbf{g})(\mathbf{x}) = \langle \bigvee\{\psi'_{n+m,n}(g_{n+m}(x_{n+m})) \mid m \in \omega\}\rangle_n \in Y$, i.e. for all n

(11) $\{\psi'_{n+m,n}(g_{n+m}(x_{n+m})) \mid m \in \omega\} \subseteq Y_n$ is directed, and

(12) $\psi'_n(j(\mathbf{g})(\mathbf{x})) = j(\mathbf{g})(\mathbf{x})$

(11): It is evident that $\{\psi'_{n+m,n}(g_{n+m}(x_{n+m})) \mid m \in \omega\} \subseteq Y_n$. By

(13)
$$\begin{aligned}
&\psi'_{n+m+1,n}(g_{n+m+1}(x_{n+m+1})) = \\
&\quad \geq \psi'_{n+m,n}(\psi'_{n+m}(g_{n+m+1}(\phi_{n+m}(\psi_{n+m}(x_{n+m+1}))))) \qquad (\phi \circ \psi \leq 1) \\
&\quad = \psi'_{n+m,n}(g_{n+m}(\psi_{n+m}(x_{n+m+1}))) \qquad (\mathbf{g} \in Z) \\
&\quad = \psi'_{n+m,n}(g_{n+m}(x_{n+m})) \qquad (\mathbf{x} \in X)
\end{aligned}$$

we see that $\{\psi'_{n+m,n}(g_{n+m}(x_{n+m})) \mid m \in \omega\}$ is directed.

(12): $\psi'_n(\bigvee\{\psi'_{n+1+m,n+1}(g_{n+1+m}(x_{n+1+m})) \mid m \in \omega\}) =$

$= \bigvee\{\psi'_{n+1+m,n}(g_{n+1+m}(x_{n+1+m})) \mid m \in \omega\}$ (continuity)

$= \bigvee\{\psi'_{n+m,n}(g_{n+m}(x_{n+m})) \mid m \in \omega - \{0\}\}$

$= \bigvee\{\psi'_{n+m,n}(g_{n+m}(x_{n+m})) \mid m \in \omega\}$ (by (13), taking m:=0),

so we are done.

(9): Let $\mathbf{g}=\langle g_n\rangle_n \in Z$. Now

$$\begin{aligned}
i(j(\mathbf{g})) &= i(\lambda x.\langle \bigvee\{\psi'_{n+m,n}(g_{n+m}(x_{n+m})) \mid m \in \omega\}\rangle_n) \\
&= \langle \lambda x_n.(\lambda x.\langle \bigvee\{\psi'_{n+m,n}(g_{n+m}(x_{n+m})) \mid m \in \omega\}\rangle_n (e(x_n)))_n\rangle_n \\
&= \langle \lambda x_n.(\langle \bigvee\{\psi'_{n+m,n}(g_{n+m}(e(x_n))_{n+m}) \mid m \in \omega\}\rangle_n)_n\rangle_n \\
&= \langle \lambda x_n.\bigvee\{\psi'_{n+m,n}(g_{n+m}(\phi_{n,n+m}(x_n))) \mid m \in \omega\}\rangle_n \\
&= \langle \lambda x_n.\bigvee\{g_n(x_n) \mid m \in \omega\}\rangle_n \qquad (g \in Z, \text{ ind. over } m) \\
&= \mathbf{g}.
\end{aligned}$$

(10). Let $f \in [X \to Y]$. Now

$$
\begin{aligned}
j(i(f)) &= j(\langle \lambda x_n.(f(e(x_n)))_n \rangle_n) \\
&= \lambda x.\langle \bigvee \{ \psi'_{n+m,n}((\lambda x_{n+m}.(f(e(x_{n+m})))_{n+m})(x)_{n+m}) \mid m \in \omega \} \rangle_n \\
&= \lambda x.\langle \bigvee \{ \psi'_{n+m,n}(f(e(x_{n+m})))_{n+m} \mid m \in \omega \} \rangle_n \\
&= \lambda x.\langle \bigvee \{ (f(e(x_{n+m})))_n \mid m \in \omega \} \rangle_n \qquad (f(e(x_{n+m})) \in Y, \text{ ind. over } m) \\
&= \lambda x.\langle (f(\bigvee \{ e(x_{n+m}) \mid m \in \omega \}))_n \rangle_n \qquad (\text{continuity of } f \text{ and } \lambda x.(x)_n) \\
&= \lambda x.\langle (fx)_n \rangle_n = f.
\end{aligned}
$$

ii) Straightforward. □

7.8. Definition.

Let $F \in \mathbb{F}(\mathbb{D}_p,\mathbb{D}_p)$. We define ch(F), the *chain of* F, by

$$
\begin{aligned}
&ch(F) := \langle F^n 1, F^n p \rangle_n, \text{ where} \\
&\qquad p = \langle \phi,\psi \rangle \text{ with } \phi : 1 \to F(1),\ \psi : F(1) \to 1, \\
&\qquad \phi(\bot_1) := \bot_{F(1)},\ \psi(x) := \bot_1 \text{ for all } x \in F(1).
\end{aligned}
$$

$$
1 \underset{\phi}{\overset{\psi}{\leftrightarrows}} F1 \underset{F\phi}{\overset{F\psi}{\leftrightarrows}} F^2 1 \underset{F^2\phi}{\overset{F^2\psi}{\leftrightarrows}} \cdots
$$

Fix(F), the *fixpoint of* F is defined by lim(ch(F)).

7.9. Lemma.

Let $F \in \mathbb{F}(\mathbb{D}_p,\mathbb{D}_p)$.

i) The chain of F is a chain.

ii) $Fix(F) \in Obj(\mathbb{D}_p)$; moreover $Fix(F) = \{ x \mid \forall n<\omega (x_n \in F^n 1 \wedge (F^n\psi)(x_{n+1}) = x_n) \}$.

iii) If F is continuous, then F(Fix(F)) = Fix(F).

Proof. i) It is clear that $\phi$ and $\psi$ are continuous and that $p = \langle \phi,\psi \rangle$ is a projection of F(1) on 1. With induction over n it follows that $F^n p$ is a projection of $F^{n+1}1$ on $F^n 1$.

ii) Straightforward by theorem 7.5.

iii) Follows from $F(Fix(F)) = F(\lim_n(F^n 1)) = \lim_n F(F^n 1) = \lim_n(F^{n+1}1) = \lim_n(F^n 1) = Fix(F)$, using lemma 7.2.(ii). □

7.10. Remark (in answer to a question of Henk Barendregt).

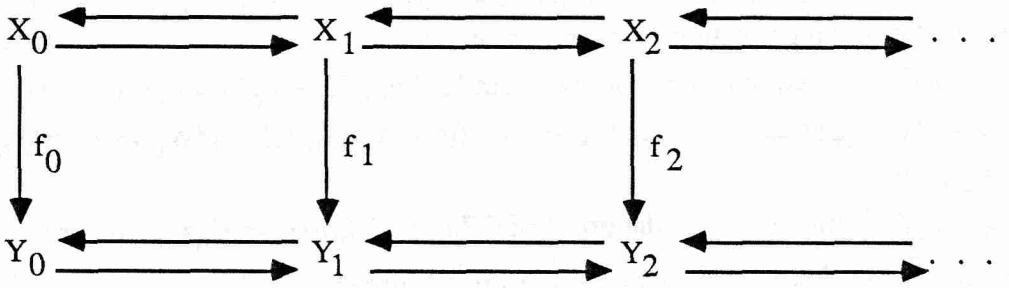
It is possible to generalize the limit construction of 7.8 by taking $\underline{2}$ instead of $\underline{1}$ (provided that $F(\underline{2}) \neq \underline{1}$). This gives more freedom for the choice of $\phi$ and $\psi$: let c be an arbitrary compact element of $F(\underline{2})$ and put

$p_c = \langle \phi_c, \psi_c \rangle$, where $\phi_c : \underline{2} \to F(\underline{2})$, $\psi_c : F(\underline{2}) \to \underline{2}$, with
$\phi_c(\bot) = \bot$, $\phi_c(\top) = c$, $\psi_c(x) = \top$ if $c \leq x$, $\psi_c(x) = \bot$ otherwise.

Now $\phi_c$ and $\psi_c$ are continuous (by lemma 3.10), and one easily checks that $p_c = \langle \phi_c, \psi_c \rangle$ is a projection.

Besides limitis of chains of domains, we also need limits of chains of continuous functions between domains. The first reason is the generalisation of the fixpoint definition to (n+1)-ary functors F, resulting in an n-ary functor Fix(F); the second is the definition of fixpoints of abstraction transformations, possibly with parameters.

7.11.Definition.
Let $\langle X_n \rangle_n = \langle X_n, \langle \phi_n, \psi_n \rangle \rangle_n$, $\langle Y_n \rangle_n = \langle Y_n, \langle \phi'_n, \psi'_n \rangle \rangle_n$ be two chains. Since $\to$ is continuous, we have $\lim_n[X_n \to Y_n] \cong [\lim_n X_n \to \lim_n Y_n]$, so an element $\mathbf{f} = \langle f_n \rangle_n \in \lim_n[X_n \to Y_n]$ corresponds to $j(\mathbf{f}) \in [\lim_n X_n \to \lim_n Y_n]$, with j as defined in the proof of 7.7.(i). To distinguish between these two representations, we call $\mathbf{f}$ an *arrow chain* and write $\lim_n f_n$ for $j(\mathbf{f})$, the *limit of* $\mathbf{f}$.




If also

$$\phi'_n \circ f_n = f_{n+1} \circ \phi_n \quad \text{for all } n,$$

then we call $\mathbf{f}$ a *strong arrow chain.*
If $\mathbf{f} = \langle f_n \rangle_n \in \lim_n[X_n \to Y_n]$ and $\mathbf{g} = \langle g_n \rangle_n \in \lim_n[Y_n \to X_n]$ are strong arrow chains with

$$f_n \circ g_n \leq 1_{Yn},\ g_n \circ f_n = 1_{Xn} \quad \text{for all } n,$$

(so $p_n = \langle f_n, g_n \rangle$ is a projection of $Y_n$ on $X_n$ for all n) then we call $\mathbf{p} = \langle p_n \rangle_n$ a *projection chain* and write $\lim_n p_n$ for $\langle j(\mathbf{f}), j(\mathbf{g}) \rangle$, the limit of $\mathbf{p}$.

We state some properties of arrow limits and chains.

7.12. Lemma.

i) Let $\mathbf{f}$ be given with $f_n \in [X_n \to Y_n]$ for all n. Then

$$\mathbf{f} \text{ is an arrow chain} \Leftrightarrow f_n = \psi'_n \circ f_{n+1} \circ \phi_n \text{ for all n.}$$
$$\mathbf{f} \text{ is a strong arrow chain} \Leftrightarrow \phi'_n \circ f_n = f_{n+1} \circ \phi_n \text{ for all n.}$$

ii) Let $f := \lim_n f_n$ be the limit of an arrow chain as above. Then

$$f(\langle x_n \rangle_n) = \langle \bigvee \{ \psi'_{n+m,n}(f_{n+m}(x_{n+m})) \mid m \in \omega \} \rangle_n;$$

moreover, if $f_n \circ \psi_n = \psi'_n \circ f_{n+1}$ for all n, then

$$f(\langle x_n \rangle_n) = \langle f_n(x_n) \rangle_n.$$

iii) If $f_n \leq g_n$ for all n, then $\lim_n f_n \leq \lim_n g_n$.

iv) $\mathbb{D}$ and $\mathbb{A}$ are closed under taking limits of arrow chains.

v) The limit of an arrow chain of abstractions is an abstraction.

vi) $\lim_n f_n \circ \lim_n g_n = \lim_n (f_n \circ g_n)$.

vii) If $f_n \leq f'_n$ for all n, then $\lim_n f_n \leq \lim_n f'_n$.

viii) The limit of a projection chain is a projection.

Proof. i) The first equivalence follows from $\mathbf{f} \in \lim_n [X_n \to Y_n] \Leftrightarrow f_n = (\phi_n \to \psi'_n) f_{n+1}$ for all n $\Leftrightarrow f_n = \psi'_n \circ f_{n+1} \circ \phi_n$ for all n; the second from $\phi'_n \circ f_n = f_{n+1} \circ \phi_n \Rightarrow f_n = \psi'_n \circ \phi'_n \circ f_n = \psi'_n \circ f_{n+1} \circ \phi_n$.

ii) By the definition of j (in the proof of 7.7.(i)). If $f_n \circ \psi_n = \psi'_n \circ f_{n+1}$ for all n, then $\psi'_{n+m,n}(f_{n+m}(x_{n+m})) = f_n(\psi_{n+m,n}(x_{n+m})) = f_n(x_n)$.

iii) straightforward.

iv) Closure for $\mathbb{D}$ is evident. As to $\mathbb{A}$, we observe that $\lim_n f_n$ is additive whenever the $f_n$ are.

v) It suffices to show, for arrow chains $\langle f_n \rangle_n$:

(14) if all $f_n$ are $\perp$-unique, then so is $\lim_n f_n$;

(15) if all $f_n$ are compactness-preserving, then so is $\lim_n f_n$.

We prove (14), (15), with $f := \lim_n f_n$.

(14) Assume all $f_n$ $\perp$-unique. Then $f\perp = \langle \bigvee \{ \psi_{n+m,n}(f_{n+m}(\perp)) \mid m \in \omega \} \rangle_n = \langle \bigvee \{ \psi_{n+m,n}(\perp) \mid m \in \omega \} \rangle_n = \langle \bigvee \{ \perp \mid m \in \omega \} \rangle_n = \perp$. On the other hand, if $f(\langle x_n \rangle_n) = \perp$, then $\bigvee \{ \psi_{n+m,n}(f_{n+m}(x_{n+m})) \mid m \in \omega \} = \perp$ for all n, so (m:=0) $f_n(x_n) = \perp$ for all n, hence $\langle x_n \rangle_n = \perp$ (for the $f_n$ are $\perp$-unique).

(15) Assume that all $f_n$ are compactness-preserving. Let $\langle x_n \rangle_n$ be compact, then (by lemma 7.4.(vi)) $\langle x_n \rangle_n = e(x_p)$ with $x_p \in X_p$ compact for some p. Now $f\langle x_n \rangle_n = \langle y_n \rangle_n$, where

$$
\begin{aligned}
y_n &= \bigvee\{\psi'_{n+m,n}(f_{n+m}(e_{n+m}(x_p))) \mid m\in\omega\} \\
&= \bigvee\{\psi'_{n+m,n}(f_{n+m}(e_{n+m}(x_p))) \mid m > |n-p|\} \quad \text{(by (13) in 7.7.(i))} \\
&= \bigvee\{\psi'_{n+m,n}(f_{n+m}(\phi_{p,n+m}(x_p))) \mid m > |n-p|\} \quad \text{(definition of } e\text{)} \\
&= \bigvee\{\psi'_{n+m,n}(\phi'_{p,n+m}(f_p(x_p))) \mid m > |n-p|\} \quad (\langle f_n\rangle_n \text{ is an arrow chain}) \\
&= \bigvee\{e_n(f_p(x_p)) \mid m > |n-p|\} = e_n(f_p(x_p)),
\end{aligned}
$$

so $\langle y_n\rangle_n = \langle e_n(f_p(x_p))\rangle_n = e(f_p(x_p))$, which is compact (for the $f_n$ preserve compactness).
vi) By (ii), we have

$$
\begin{aligned}
&((\lim_n f_n \circ \lim_n g_n)x)_n = \\
&= \bigvee\{\psi'_{n+m,n}(f_{n+m}(\bigvee\{\psi''_{n+m+p,n+m}(g_{n+m+p}(x_{n+m+p})) \mid p\in\omega\})) \mid m\in\omega\} \\
&= \bigvee\{\psi'_{n+m,n}(f_{n+m}(\psi''_{n+m+p,n+m}(g_{n+m+p}(x_{n+m+p})))) \mid m,p\in\omega\}
\end{aligned}
$$

(the last equality holds by continuity), and

$$
(\lim_n(f_n \circ g_n)x)_n = \bigvee\{\psi'_{n+m,n}(f_{n+m}(g_{n+m}(x_{n+m}))) \mid m\in\omega\},
$$

so we have to show $\bigvee\{t_{m,n,p} \mid m,p\in\omega\} = \bigvee\{t_{m,n,0} \mid m\in\omega\}$, where $t_{m,n,p} := \psi'_{n+m,n}(f_{n+m}(\psi''_{n+m+p,n+m}(g_{n+m+p}(x_{n+m+p}))))$. This is true if $t_{m,n,p} \le t_{m+p,n,0}$, and this follows from

$$
f_{n+m} \circ \psi''_{n+m+p,n+m} \le \psi'_{n+m+p,n+m} \circ f_{n+m+p}
$$

which is proved with induction over p, using $f_k \circ \psi''_k \le \psi'_k \circ f_{k+1}$. This last inequality holds because of $f_k = \psi'_k \circ f_{k+1} \circ \phi''_k \Rightarrow f_k \circ \psi''_k = \psi'_k \circ f_{k+1} \circ \phi''_k \circ \psi''_k \le \psi'_k \circ f_{k+1}$.
vii) Easy, by the definition of j.
viii) Follows directly from (vi) and (vii). []

Now we turn to (n+1)-ary functors. First some notation: if $M : X*Y \to Z$ is some (possibly partial) mapping and $x\in X$, then $M_x := \lambda y.M(\langle x,y\rangle)$.

7.13. Lemma.
i) If $F\in \mathbb{F}(\mathbb{D}^m*\mathbb{D},\mathbb{D})$ and $f\in \mathrm{Obj}(\mathbb{D}^m)$ (i.e. $f = 1_X$ for some $X\in \mathrm{Obj}(\mathbb{D}^m)$), then $F_f\in \mathbb{F}(\mathbb{D},\mathbb{D})$; moreover, if F is continuous, then so is $F_f$. Idem for $\mathbb{D}_p$ or $\mathbb{A}_p$ instead of $\mathbb{D}$.
ii) If $H : \mathbb{D}^m*\mathbb{D} \to \mathbb{D}$ (possibly partial), $H : F \Rightarrow G$ and $f\in \mathrm{Ar}(\mathbb{D}^m)$ is an abstraction, then $H_f : F_{\mathrm{dom}(f)} \Rightarrow G_{\mathrm{cod}(f)}$.
Proof. i) Easy, $f\in \mathrm{Obj}(\mathbb{D}^m)$ is required to get $X\in \mathrm{Obj}(\mathbb{D}) \Rightarrow F_f(X)\in \mathrm{Obj}(\mathbb{D})$.
ii) Also easy, using that $(\langle 1_{\mathrm{dom}(f)},1_{\mathrm{dom}(f)}\rangle, \langle 1_{\mathrm{cod}(f)},1_{\mathrm{cod}(f)}\rangle,f,f)$ is an abstraction diagram whenever f is an abstraction. []

7.14. Definition.

Let $F \in \mathbb{F}(\mathbb{D}_p{}^m * \mathbb{D}_p, \mathbb{D}_p)$, $p \in Ar(\mathbb{D}_p{}^m)$. Then $F_{dom(p)}, F_{cod(p)} \in \mathbb{F}(\mathbb{D}_p, \mathbb{D}_p)$ and we have the chains $<X_n>_n = <F_{dom(p)}{}^n(1)>_n$, $<Y_n>_n = <F_{cod(p)}{}^n(1)>_n$ of $F_{dom(p)}$ and $F_{cod(p)}$, respectively.

Now the *projection chain* $\mathbf{p} = <\mathbf{f},\mathbf{g}> = <f_n,g_n>_n \in \lim_n[X_n \to Y_n] \times \lim_n[Y_n \to X_n]$ *of* F *in* p, alias the *fixpoint* $Fix(F_p) \in [\lim_n X_n \to \lim_n Y_n] \times [\lim_n Y_n \to \lim_n X_n]$ *of* F *in* p, is defined by

$$<f_n,g_n> := F_p{}^n(1), \text{ for all } n.$$

We also define the *fixpoint functor* Fix(F) by

$$Fix(F) := \lambda p.Fix(F_p).$$

Idem for $\mathbb{A}_p$ instead of $\mathbb{D}_p$.

7.15. Lemma.

These definitions are correct, i.e. $\mathbf{p} \in \lim_n[X_n \to Y_n] \times \lim_n[Y_n \to X_n]$ is a projection chain and $Fix(F) \in \mathbb{F}(\mathbb{D}_p, \mathbb{D}_p)$.

Proof. $p_n = F_p{}^n(1) \in [F_{dom(p)}{}^n(1) \to F_{cod(p)}{}^n(1)] \times [F_{cod(p)}{}^n(1) \to F_{dom(p)}{}^n(1)]$ is proved with induction over n; idem for $q'_n \circ p_n = p_{n+1} \circ q_n$, i.e.

$$F_{cod(p)}{}^n(q') \circ F_p{}^n(1) = F_p{}^{n+1}(1) \circ F_{dom(p)}{}^n(q) \tag{16}$$

where

$$q = <\phi,\psi> \in [1 \to dom(F(p,1))] \times [dom(F(p,1)) \to 1]$$
$$\text{with } \phi(\bot) = \bot, \psi(x) = \bot \text{ for all } x \in dom(F(p,1)),$$
$$q' = <\phi',\psi'> \in [cod(F(p,1)) \to 1] \times [cod(F(p,1)) \to 1]$$
$$\text{with } \phi'(\bot) = \bot, \psi'(x) = \bot \text{ for all } x \in cod(F(p,1)).$$

The base step is

$$q' \circ 1 = F(p,1) \circ q$$

and this follows from the fact that q, q' and F(p,1) are projections, hence their components are strict (lemma 6.2.(ii)). The induction step,

$$F(cod(p),F_{cod(p)}{}^n(q')) \circ F(p,F_p{}^n(1)) = F(p,F_p{}^{n+1}(1)) \circ F(dom(p),F_{dom(p)}{}^n(q))$$

follows directly from (16), $cod(p) \circ p = p \circ dom(p)$ and the fact that F is a functor.

To see that $Fix(F) \in \mathbb{F}(\mathbb{D}_p, \mathbb{D}_p)$, we observe $(Fix(F))(p) \circ (Fix(F))(p') = Fix(F_p) \circ Fix(F_{p'}) = \lim_n F_p{}^n(1) \circ \lim_n F_{p'}{}^n(1) = \lim_n (F_p{}^n(1) \circ F_{p'}{}^n(1)) = \lim_n (F_{p \circ p'}{}^n(1)) = Fix(F_{p \circ p'}) = (Fix(F))(p \circ p')$, using 7.12.(vi) for the third equality. □

Before we can show that continuity is preserved under applying Fix, we need a lemma on the permutation of limits.

7.16. Lemma.
Let ${}_mX_n$, ${}_mp'_n$, ${}_{m'}p_n$ $(m,n \in \omega)$ be given with
a) $\langle {}_mX_n, {}_mp'_n \rangle_n$ is a chain, for every $m \in \omega$;
b) $\langle {}_mX_n, {}_mp'_n \rangle_m$ is a chain, for every $n \in \omega$;
c) ${}_{m'}p_{n+1} \circ {}_mp'_n = {}_{m+1}p'_n \circ {}_{m'}p_n$.
Then $\lim_m(\lim_n {}_mX_n) = \lim_n(\lim_m {}_mX_n)$.
Proof. By (c), we have ${}_{m'}\psi_{n+1} \circ {}_m\psi'_n = {}_{m+1}\psi'_n \circ {}_{m'}\psi_n$ for all $m,n \in \omega$, so
$\lim_n({}_{m'}\psi_n)\langle {}_{m+1}x_n \rangle_n = \langle \bigvee \{ {}_m\psi'_{n+p,n}({}_{m'}\psi_{n+p}({}_{m+1}x_{n+p})) \mid p \in \omega \} \rangle_n = \langle {}_{m'}\psi_n({}_{m+1}x_n) \rangle_n$
and analogously $\lim_m({}_m\psi'_n)\langle {}_mx_{n+1} \rangle_m = \langle {}_m\psi'_n({}_mx_{n+1}) \rangle_m$ So we get

$$
\begin{aligned}
&\lim_m(\lim_n {}_mX_n) = \\
&= \{ \langle\langle {}_mx_n \rangle_n \rangle_m \mid \forall m(\langle {}_mx_n \rangle_n \in \lim_n {}_mX_n \wedge \lim_n({}_{m'}\psi_n)\langle {}_{m+1}x_n \rangle_n = \langle {}_mx_n \rangle_n) \} \\
&= \{ \langle\langle {}_mx_n \rangle_n \rangle_m \mid \forall mn({}_m\psi'_n({}_mx_{n+1}) = {}_mx_n \wedge {}_{m'}\psi_n({}_{m+1}x_n) = {}_mx_n) \} \\
&= \{ \langle\langle {}_mx_n \rangle_m \rangle_n \mid \forall n(\langle {}_mx_n \rangle_n \in \lim_m {}_mX_n \wedge \lim_m({}_m\psi'_n)\langle {}_mx_{n+1} \rangle_m = \langle {}_mx_n \rangle_m) \} \\
&= \lim_n(\lim_m {}_mX_n).
\end{aligned}
$$
□

Now we can prove that Fix preserves continuity.

7.17. Lemma.
If F is continuous, then so is Fix(F).
Proof. Let $\langle X_n \rangle_n = \langle X_n, p_n \rangle_n$ be some chain. We have

$$
\begin{aligned}
\lim_n(Fix(F)(X_n)) &= \lim_n(\lim_m F_{X_n}{}^m(1)) \\
&= \lim_m(\lim_n F_{X_n}{}^m(1)) && \text{(step I)} \\
&= \lim_m(F_{\lim_n X_n}{}^m(1)) && \text{(step II)} \\
&= Fix(F)(\lim_n X_n)
\end{aligned}
$$

so we only have to justify step I and step II to obtain the continuity of Fix(F).
Step I: this is an application of lemma 7.16, so we have to verify:

(17) $\langle F_{X_n}{}^m(\underline{1}), F_{X_n}{}^m(q_n) \rangle_m$ is a chain, for all n;
(18) $\langle F_{X_n}{}^m(\underline{1}), F_{p_n}{}^m(\underline{1}) \rangle_n$ is a chain, for all m;
(19) $F_{X(n+1)}{}^m(q_{n+1}) \circ F_{p_n}{}^m(\underline{1}) = F_{p_n}{}^{m+1}(\underline{1}) \circ F_{X_n}{}^m(q_n)$, for all m,n.

where

$$q_n = \langle \phi_n, \psi_n \rangle \text{ with}$$
$$\phi_n : 1 \to F(X_n, \underline{1}) \text{ satisfying } \phi_n(\bot) = \bot,$$
$$\psi'_m : F(X_n, \underline{1}) \to 1, \text{ satisfying } \psi_n(x) = \bot \text{ for all } x \in F(X_n, \underline{1}).$$

Now (17) holds since $q_n$ is a projection and $F_{Xn}$ is a functor, (18) since $\langle X_n, p_n \rangle_n$ is a chain and $\lambda f.F_f^m(\underline{1})$ is a functor (induction over m); (19) follows with induction over m, with at the base $q_{n+1} \circ \underline{1} = F(p_n, \underline{1}) \circ q_n$, a direct consequence of the strictness of the components of $q_{n+1}$, $\underline{1}$, $F(p_n, \underline{1})$ and $q_n$ (by lemma 6.2.(ii)).
Step II: follows with induction over m. The base is trivial (1=1), for the induction step we argue as follows:

$$\begin{aligned}
\lim_n F_{Xn}^{m+1}(1) &= \lim_n F_{Xn}(F_{Xn}^m(1)) \\
&= \lim_n F(X_n, F_{Xn}^m(1)) \\
&= F(\lim_n X_n, \lim_n F_{Xn}^m(1)) \quad \text{(continuity of } F\text{)} \\
&= F(\lim_n X_n, F_{\lim n\, Xn}^m(1)) \quad \text{(induction hypothesis)} \\
&= F_{\lim n\, Xn}^{m+1}(1).
\end{aligned}$$

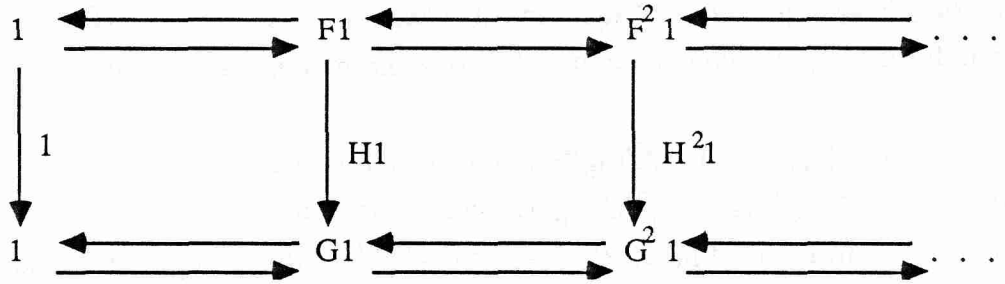
This ends the proof. []

Finally we consider fixpoints of abstraction transformations, beginning with the simple case without parameters.

7.18. Definition.
Let $F \in \mathbb{F}(\mathbb{D}_p, \mathbb{D}_p)$, $G \in \mathbb{F}(\mathbb{A}_p, \mathbb{A}_p)$. (Then $\langle F^n(1) \rangle_n$, $\langle G^n(1) \rangle_n$ are chains, by 7.9.)
For $H : F \Rightarrow G$, we define

$$\langle H^n(1) \rangle_n \in \lim_n [F^n(1) \to G^n(1)]$$

as the *arrow chain of* H , alias the *fixpoint* $\mathrm{Fix}(H) \in [\lim_n F^n(1) \to \lim_n G^n(1)]$ *of* H.



7.19. Lemma.

i) $\langle H^n(1)\rangle_n$, as defined above, is a strong arrow chain.

ii) Fix(H) is an abstraction.

Proof. i) We have to show

$$G^n(\phi')\circ H^n(1) = H^{n+1}(1)\circ F^n(\phi) \quad \text{for all } n,$$

i.e. $(F^n(\phi),G^n(\phi'),H^n(1), H^{n+1}(1))$ is an abstraction diagram. This follows with induction over n. The base step is

$$\phi'\circ 1 = H(1)\circ\phi$$

and follows from the fact that H preserves abstractions, so H(1) is strict. For the induction step we use $H : F \Rightarrow G$.

ii) Follows directly from lemma 7.12.(v). []

7.20. Definition.

Let $F\in \mathbb{F}(\mathbb{D}_p{}^{m}{*}\mathbb{D}_p,\mathbb{D}_p)$, $G\in \mathbb{F}(\mathbb{A}_p{}^{m}{*}\mathbb{A}_p,\mathbb{A}_p)$, $f\in Ar(\mathbb{D}^m)$ an abstraction. Then we have the chains $\langle X_n\rangle_n=\langle F_{dom(f)}{}^n(1)\rangle_n$ of $F_{dom(f)}$ and $\langle Y_n\rangle_n=\langle G_{cod(f)}{}^n(1)\rangle_n$ of $G_{cod(f)}$.
If $H : F \Rightarrow G$, the *arrow chain*

$$\mathbf{h} = \langle h_n\rangle_n\in \lim_n[X_n\to Y_n]$$

*of* H *in* f, alias the *fixpoint* $Fix(H_f)\in [\lim_n X_n\to\lim_n Y_n]$ *of* H *in* f, is defined by

$$h_n := H_f{}^n(1), \text{ for all } n.$$

We also define the *fixpoint functor* Fix(H) by

$$Fix(H) := \lambda f.Fix(H_f).$$

7.21. Lemma.

If $H : F \Rightarrow G$, then $Fix(H) : Fix(F) \Rightarrow Fix(G)$.

Proof.

Assume $\phi'\circ f = g\circ\phi$, f and g abstractions. By lemma 7.13.(ii) and lemma 7.19.(ii) we see that $Fix(H)f = \langle H_f{}^n(1)\rangle_n$ and $Fix(H)g = \langle H_g{}^n(1)\rangle_n$ are also abstractions. So we only need $(Fix(G)\phi')\circ(Fix(H)f) = (Fix(H)g)\circ(Fix(F)\phi)$, i.e.

$$\lim_n G_{\phi'}{}^n(1) \circ \lim_n H_f{}^n(1) = \lim_n H_g{}^n(1) \circ \lim_n F_\phi{}^n(1);$$

by lemma 7.12.(vi) this comes down to

$$G_{\phi'}{}^n(1) \circ H_f{}^n(1) = H_g{}^n(1) \circ F_\phi{}^n(1) \quad \text{for all } n.$$

We prove this with induction over n. For n=0 this is trivial; the induction step requires

$$G(\phi', G_{\phi'}{}^n(1)) \circ H(f, H_f{}^n(1)) = H(g, H_g{}^n(1)) \circ F(\phi, F_\phi{}^n(1))$$

and this follows from the induction hypothesis, $\phi' \circ f = g \circ \phi$ and $H : F \Rightarrow G$. []

## § 8. TWO INTERPRETATIONS AND ABSTRACTION.

In this section we define two interpretations of the language POLYREC. The first, called the *concrete* interpretation and denoted by CI, is in the category $\mathbb{D}$ of domains. It is, in a sense, the intended interpretation of the language. We also introduce the *abstract* interpretation, denoted by AI, which resides in the category $\mathbb{A}$ of algebraic complete lattices.
Then a mapping abs (abstraction) between the range of CI and that of AI is presented. It embodies the notion of *abstraction*, used for strictness analysis.

8.1. Definition.
A $\mathbb{D}_p$-*assignment* a is a mapping of type variables on arrows of $\mathbb{D}_p$. If $\alpha$ is a type variable and f is an arrow of $\mathbb{D}_p$, then $a':=a[\alpha\rightarrow f]$ is the assignment defined by $a'(\alpha)=f$, $a'(\beta)=a(\beta)$ if $\beta$ is a type variable different from $\alpha$.
a is a *normal* assignment if it is a mapping on $Obj(\mathbb{D}_p)$.

We start with the definition of the concrete interpretation CI for types and type functions.

8.2. Definition.
Let a be a $\mathbb{D}_p$-assignment. Then:

$$CI(\alpha,a) = a(\alpha)$$
$$CI(N,a) = N$$
$$CI(B,a) = B$$
$$CI(\times,a) = \times$$
$$CI(+,a) = +$$
$$CI(\rightarrow,a) = \rightarrow$$
$$CI(\Phi\tau_1\ldots\tau_n,a) = CI(\Phi,a)(CI(\tau_1,a))\ldots(CI(\tau_n,a))$$
$$CI(\delta\alpha_1\ldots\alpha_n.\tau,a) = \lambda f_1\ldots f_n \in Ar(\mathbb{D}_p).CI(\tau,a[\alpha_1\rightarrow f_1,\ldots,\alpha_n\rightarrow f_n])$$
$$CI(\mu\Phi,a) = Fix(CI(\Phi,a))$$

If the expression e of the language contains no free (type or term) variables, then the value of CI(e,a) does not depend on a : in such cases we shall write CI(e) for CI(e,a). A more conventional notation for CI(e) is $[\![e]\!]$ or $[\![e]\!]_c$.

8.3. Lemma.
Let a be a normal $\mathbb{D}_p$-assignment, $\tau \in \Theta$, $\Phi \in \Theta^n\rightarrow\Theta$. Then:

$$CI(\tau,a) \in \mathbb{D},$$
$$CI(\Phi,a) \in \mathbb{CF}(\mathbb{D}_p{}^n,\mathbb{D}_p).$$

Proof. Let $* := \langle\alpha_1,\ldots,\alpha_n\rangle$ be some sequence of type variables, $\tau \in \Theta$ and $\Phi \in \Theta^m \to \Theta$. Then we define

$$\tau^* := \delta\alpha_1\ldots\alpha_n.\tau,$$
$$\Phi^* := \delta\alpha_1\ldots\alpha_n\beta_1\ldots\beta_m.\Phi\beta_1\ldots\beta_m \quad (\{\alpha_1\ldots\alpha_n\} \text{ and } \{\beta_1\ldots\beta_m\} \text{ disjoint}).$$

With simultaneous induction over the complexity of $\tau \in \Theta$ and $\Phi \in \Theta^m \to \Theta$, we prove, for arbitrary sequences of type variables $*$ with length n:

(1) $\quad CI(\tau^*,a) \in \mathbb{CF}(\mathbb{D}_p{}^n,\mathbb{D}_p)$,

(2) $\quad CI(\Phi^*,a) \in \mathbb{CF}(\mathbb{D}_p{}^{n+m},\mathbb{D}_p)$;

from (1), (2) the lemma follows (take $* := \langle\rangle$).

We treat some typical cases (the others are analogous or trivial).

$\tau = \alpha$, $* = \langle\alpha\rangle$: $CI(\alpha^*,a) = CI(\lambda\alpha.\alpha,a) = \lambda f.CI(\alpha,a[\alpha\to f]) = \lambda f.f \in \mathbb{CF}(\mathbb{D}_p,\mathbb{D}_p)$.

$\tau = N$, $* = \langle\rangle$: $CI(N,a) = N \in \mathbb{D}_p = \mathbb{CF}(\mathbb{D}_p{}^0,\mathbb{D}_p)$.

$\Phi = \times, +$ or $\to$ : (2) follows directly from lemma 7.7.(i) ($\times$, $+$ and $\to$ are continuous).

$\tau = \Phi\tau_1\ldots\tau_n$ : (1) follows from lemma 7.7.(ii) (continuous functors are closed under composition) and induction hypothesis (1) for $\tau_1,\ldots,\tau_n$ and (2) for $\Phi$.

$\Phi = \delta\beta_1\ldots\beta_n.\tau$ : (2) follows directly from induction hypothesis (1) for $\tau$.

$\tau = \mu\Phi$, $* = \langle\alpha\rangle$: by induction hypothesis (2) for $\Phi$, we have $CI(\Phi^*,a) \in \mathbb{CF}(\mathbb{D}_p{}^*\mathbb{D}_p,\mathbb{D}_p)$, so

$$\begin{aligned} CI(\mu\Phi^*,a) &= CI(\lambda\alpha.\mu\Phi,a) \\ &= \lambda f.CI(\mu\Phi,a[\alpha\to f]) \\ &= \lambda f.Fix(CI(\Phi,a[\alpha\to f])) \\ &= Fix(\lambda f.(CI(\Phi,a[\alpha\to f]))) \\ &= Fix(CI(\lambda\alpha\beta.\Phi(\beta),a)) \\ &= Fix(CI(\Phi^*,a)) \\ &\in \mathbb{CF}(\mathbb{D}_p,\mathbb{D}_p) \quad \text{(by lemma 7.17).} \end{aligned}$$

[]

Before we extend the concrete interpretation to (polymorphic) terms, we extend the assignments. The naive idea is: let a map type variables on objects of a category and term variables on elements of these objects, such that

(*) $\quad a(x) \in a(\mathbf{type}(x))$.

This is too simple, however: it should be possible to modify a to e.g. $a[\alpha\to D]$, but then (*) may be violated, since the type of a term variable may contain type variables. This problem is dealt with in the following definition.

8.4. Definition.

i) If $F \in \mathbb{F}(\mathbb{B},\mathbb{C})$, the objects of $\mathbb{C}$ are sets and $f : Obj(\mathbb{B}) \to \cup Obj(\mathbb{C})$ satisfies $f(B) \in F(B)$ for all $B \in Obj(\mathbb{B})$, then we call $f$ a *choice function associated with* $F$, notation $f \varepsilon F$.

ii) If $a$ is a normal $\mathbb{D}_p$-assignment which maps term variables on choice functions of $\mathbb{D}_p$ in such a way that

$$a(x) \varepsilon CI(\mathbf{type}(x)^*),$$

where $^* = <\alpha_1,\ldots,\alpha_n>$ are the free type variables in **type**(x) (so **type**(x)* is closed), then we call $a$ an *extended* $\mathbb{D}_p$-assignment. Furthermore we put

$$a_*(x) := a(x)(a(\alpha_1),\ldots,a(\alpha_n)),$$

so $a_*(x) \in CI(\mathbf{type}(x),a)$.

iii) If $a$ is an extended $\mathbb{D}_p$-assignment, $x$ is a term variable and $f \varepsilon CI(\mathbf{type}(x)^*,a)$ (* as above), then $a':=a[x\to f]$ is the extended assignment defined by $a'(x)=f$, $a'(y)=a(y)$ if $y$ is a term variable different from $x$.

iv) If $d \in CI(\mathbf{type}(x),a)$, $\mathbf{type}(x) = \tau[\alpha_1,\ldots,\alpha_n]$, then we put $d^x := \lambda D_1 \ldots D_n \in Obj(\mathbb{D}_p).d$.

Fact. If $a$ is an extended $\mathbb{D}_p$-assignment satisfying $d \in a(\mathbf{type}(x))$, then $d^x \varepsilon CI(\mathbf{type}(x)^*)$, so $a[x\to d^x]$ is welldefined.

8.5. Definition.

Let $a$ be an extended $\mathbb{D}_p$-assignment. Then we define:

$CI(x,a) = a_*(x)$
$CI(0,a) = 0$
$CI(S,a) = S$
$CI(plus,a) = +$
$CI(times,a) = .$
$CI(eq,a) = eq$
$CI(true,a) = true$
$CI(false,a) = false$
$CI(not,a) = not$
$CI(and,a) = and$
$CI(or,a) = or$
$CI(\bot,a) = \lambda D \in Obj(\mathbb{D}_p).\bot$
$CI(ite,a) = ite$
$CI(fix,a) = fix$
$CI(pair,a) = pair$

$CI(p0,a) = p0$
$CI(p1,a) = p1$
$CI(in0,a) = in0$
$CI(in1,a) = in1$
$CI(case,a) = case$
$CI(st,a) = CI(s,a)CI(t,a)$
$CI(\lambda x{:}\tau.t,a) = \lambda d \in CI(\tau,a).CI(t,a[x \to d^x])$
$CI(p\tau_1 \ldots \tau_n,a) = CI(p,a)(CI(\tau_1,a))\ldots(CI(\tau_n,a))$
$CI(\Lambda\alpha_1 \ldots \alpha_n.t,a) = \lambda D_1 \ldots D_n \in Obj(\mathbb{D}_p).CI(t,a[\alpha_1 \to D_1,\ldots,\alpha_n \to D_n])$

8.6. Lemma.
Let a be a extended $\mathbb{D}_p$-assignment, $t \in T$, $p \in \Theta^n \to T$, $D_1,\ldots,D_n \in Obj(\mathbb{D})$. Then:

$CI(t,a) \in CI(\mathbf{type}(t),a)$.
$CI(p,a)D_1 \ldots D_n \in CI(\mathbf{type}(p),a)D_1 \ldots D_n$.

Proof. Let $\# := \langle \alpha_1,\ldots,\alpha_n,x_1,\ldots,x_k \rangle$ be a sequence of type and term variables, $t \in T$ with its free varables among $x_1,\ldots,x_k$ and $\mathbf{type}(x_i) = \tau_i$ $(i = 1,\ldots,k)$. Then we define

$$t^{\#} := \Lambda\alpha_1 \ldots \alpha_n.\lambda x_1{:}\tau_1 \ldots x_k{:}\tau_k.t;$$

Now let $* := \langle \alpha_1,\ldots,\alpha_n \rangle$ be a sequences of type variables, $p \in \Theta^m \to T$. Then we define

$$\Phi* := \Lambda\alpha_1 \ldots \alpha_n\beta_1 \ldots \beta_m.p\beta_1 \ldots \beta_m$$

here $\{\alpha_1 \ldots \alpha_n\}$ and $\{\beta_1 \ldots \beta_m\}$ are disjoint.
With simultaneous induction over the complexity of $t \in T$ and $p \in \Theta^m \to T$, we prove, for arbitrary sequences of type and term variables $*$ and $\#$ as defined above:

(3) $CI(t^{\#},a) \varepsilon CI(\mathbf{type}(t^{\#}),a)$,
(4) $CI(p*,a) \varepsilon CI(\mathbf{type}(p*),a)$;

from (3), (4) the lemma follows (take $\# := \langle\rangle$).
$t = x$, $\mathbf{type}(x) = \tau = \tau[\alpha]$, $\# = \langle\alpha,x\rangle$: now $CI(x^{\#},a) = CI(\Lambda\alpha.\lambda x{:}\alpha.x,a) = \lambda D.\lambda d \in D.d \varepsilon \lambda D.D$
$= CI(\lambda\alpha.\alpha \to \alpha,a) = CI(\mathbf{type}(x)^{\#},a)$.
The other cases are straightforward. []

8.7. Lemma.
The interpretation CI is sound w.r.t. the axiomatics of POLYREC.
Proof. We only check the two type axioms listed in 2.3:

(5) $(\delta\alpha_1\ldots\alpha_n.\tau)\tau_1\ldots\tau_n = \tau[\alpha_1:=\tau_1,\ldots,\alpha_n:=\tau_n]$

(6) $(\delta\alpha.\tau)(\mu(\delta\alpha.\tau)) = \mu(\delta\alpha.\tau)$

the others are easy.

(5): let, for simplicity, n=1, then we have to show $CI((\delta\alpha.\tau)\sigma,a) = CI(\tau[\alpha:=\sigma],a)$ and this follows from $CI(\tau,a[\alpha\to CI(\sigma,a)]) = CI(\tau[\alpha:=\sigma],a)$, which is proved with induction over the complexity of $\tau$.

(6): now we have to show, for $\Phi \in \Theta\to\Theta$: $CI(\Phi(\mu\Phi),a) = CI(\mu\Phi,a)$, i.e. $CI(\Phi,a)(Fix(CI(\Phi,a))) = Fix(CI(\Phi,a))$, and this follows by lemma 7.9.(iii) and lemma 8.3. □

Now we turn to the abstract interpretation AI.

8.8. Definition.

Let a be an $\mathbb{A}_p$-assignment. Then we define:

$AI(\alpha,a) = a(\alpha)$

$AI(N,a) = \underline{2}$

$AI(B,a) = \underline{2}$

$AI(\times,a) = \times$

$AI(+,a) = +_c$

$AI(\to,a) = \to$

$AI(\Phi\tau_1\ldots\tau_n,a) = AI(\Phi,a)(AI(\tau_1,a))\ldots(AI(\tau_n,a))$

$AI(\delta\alpha_1\ldots\alpha_n.\tau,a) = \lambda f_1\ldots f_n \in Ar(\mathbb{A}_p).AI(\tau,a[\alpha_1\to f_1,\ldots,\alpha_n\to f_n])$

$AI(\mu\Phi,a) = fix(AI(\Phi,a))$

Let a be an extended $\mathbb{A}_p$-assignment. Then:

$AI(x,a) = a_*(x)$

$AI(0,a) = \mathsf{T}$

$AI(S,a) = \lambda x\in\underline{2}.x$

$AI(plus,a) = \lambda xy\in\underline{2}.x\wedge y$

$AI(times,a) = \lambda xy\in\underline{2}.x\wedge y$

$AI(eq,a) = \lambda xy\in\underline{2}.x\wedge y$

$AI(true,a) = \mathsf{T}$

$AI(false,a) = \mathsf{T}$

$AI(not,a) = \lambda x\in\underline{2}.x$

$AI(and,a) = \lambda xy\in\underline{2}.x\wedge y$

$AI(or,a) = \lambda xy\in\underline{2}.x\wedge y$

$AI(\bot,a) = \lambda A\in Obj(\mathbb{A}_p).\bot$

$AI(ite,a) = ite_C$
$AI(fix,a) = fix$
$AI(pair,a) = pair$
$AI(p0,a) = p0$
$AI(p1,a) = p1$
$AI(in0,a) = in0_C$
$AI(in1,a) = in1_C$
$AI(case,a) = case_C$
$AI(st,a) = AI(s,a)AI(t,a)$
$AI(\lambda x{:}\tau.t,a) = \lambda d \in AI(\tau,a).AI(t,a[x \to d])$
$AI(p\tau_1 \ldots \tau_n,a) = AI(p,a)(AI(\tau_1,a))\ldots(AI(\tau_n,a))$
$AI(\Lambda\alpha_1 \ldots \alpha_n.t,a) = \lambda A_1 \ldots A_n \in Obj(\mathbb{A}_p).AI(t,a[\alpha_1 \to A_1,\ldots,\alpha_n \to A_n])$

8.9. Lemma.
Let a be an extended $\mathbb{A}_p$-assignment, $\tau \in \Theta$, $\Phi \in \Theta^n \to \Theta$, $t \in T$, $p \in \Theta^n \to T$. Then:

$AI(\tau,a) \in Obj(\mathbb{A}_p)$
$AI(\Phi,a) \in \mathbb{CF}(\mathbb{A}_p{}^n,\mathbb{A}_p)$
$AI(t,a) \in AI(\mathbf{type}(t),a)$
$AI(p,a) \in AI(\mathbf{type}(p),a)$

Moreover, the interpretation AI is sound w.r.t. the axiomatics of POLYREC.
Proof. As for 8.3, 8.6 and 8.7. []

Now we define the abstraction mapping abs between the concrete and the abstract interpretation. Let, from now on, a and a' be normal $\mathbb{D}_p$- resp. $\mathbb{A}_p$-assignments, and let a" be an *associated abstraction assignment*, i.e. a mapping of type variables $\alpha$ on abstractions $a''(\alpha) \in [a(\alpha) \to a'(\alpha)]$. Such a,a',a" are called *assignments triples*.

8.10. Definition.
abs is defined by:

$abs(\alpha,a'') = a''(\alpha)$
$abs(N,a'') = abs_N$
$abs(B,a'') = abs_B$
$abs(\times,a'') = \times$
$abs(+,a'') = +\!+$
$abs(\to,a'') = \twoheadrightarrow$

$abs(\delta\alpha_1 \ldots \alpha_n.\tau,a'') = \lambda f_1 \ldots f_n \in Ar(\mathbb{D}).abs(\tau,a''[\alpha_1 := f_1, \ldots, \alpha_n := f_n])$
$abs(\Phi\tau_1 \ldots \tau_n,a'') = abs(\Phi,a'')(abs(\tau_1,a'')) \ldots (abs(\tau_n,a''))$
$abs(\mu\Phi,a'') = Fix(abs(\Phi,a''))$

8.11. Facts.
i) $abs(\sigma\times\tau)<x,y> = <abs(\sigma)x,abs(\tau)y>$.
ii) $abs(\sigma+\tau)\bot = \bot$, $abs(\sigma\times\tau)<0,x> = <0,abs(\sigma)x>$, $abs(\sigma\times\tau)<1,y> = <1,abs(\tau)y>$.
iii) $abs(\sigma\to\tau)x = \lambda y \in AI(\sigma).\bigvee\{abs(\tau)(xz) \mid abs(\sigma)z \leq y\}$.

8.12. Lemma.
Let $\tau \in \Theta$, $\Phi \in \Theta^n \to \Theta$. Then:

$abs(\tau,a'') \in [CI(\tau,a) \to AI(\tau,a')]$ is an abstraction,
$abs(\Phi,a'') : CI(\Phi,a) \Rightarrow AI(\Phi,a')$.

Proof. Let $*$, $\tau^*$ and $\Phi^*$ be defined as in the proof of lemma 8.3. With simultaneous induction over the complexity of $\tau \in \Theta$ and $\Phi \in \Theta^m \to \Theta$, we prove, for arbitrary sequences of type variables $*$ with length n:

(7) $abs(\tau^*,a'') : CI(\tau^*,a) \Rightarrow AI(\tau^*,a')$,
(8) $abs(\Phi^*,a'') : CI(\Phi^*,a) \Rightarrow AI(\Phi^*,a')$;

from (7), (8) the lemma follows (take $* := <>$).
$\tau = \alpha$, $* = <\alpha>$: $abs(\alpha^*,a'') = abs(\delta\alpha.\alpha,a'') = \lambda f.abs(\alpha,a''[\alpha \to f]) = \lambda f.f$, and it is evident that $\lambda f.f : \lambda f.f \Rightarrow \lambda f.f$.
$\tau = N$, $* = <>$ : $abs^*(N,a'') = abs(N) = abs_N \in [N \to \underline{2}] = [CI(N) \to AI(N)]$, and $abs_n$ is an abstraction.
$\tau = B$ : analogously.
$\Phi = \times, +$ or $\to$ : (8) follows directly from the definition of abs and lemma 6.11.(i-iii) ($\times : \times \Rightarrow \times$, $++ : + \Rightarrow +_c$, $\twoheadrightarrow : \to \Rightarrow \to$).
$\tau = \Phi\tau_1 \ldots \tau_n$ : (7) follows from lemma 6.11.(iv) (abstraction transformations are closed under composition), induction hypothesis (7) for $\tau_1, \ldots, \tau_n$ and induction hypothesis (8) for $\Phi$.
$\Phi = \delta\beta_1 \ldots \beta_n.\tau$ : (8) follows directly from induction hypothesis (7) for $\tau$.
$\tau = \mu\Phi$: by induction hypothesis (8) for $\Phi$ and lemma 7.21. []

Before we can prove that $abs \circ CI \leq AI$, we need two inequalities concerning the behaviour of ter application and the fixpoint operator under abs.

8.13. Lemma.
i) If $x \in CI(\sigma\to\tau,a)$ and $y \in CI(\sigma,a)$, then $abs(\tau,a'')(x(y)) \leq (abs(\sigma\to\tau,a'')x)(abs(\sigma,a'')y)$; in short notation

$$abs(x(y)) \leq abs(x)(abs(y)).$$

ii) $abs((\tau\to\tau)\to\tau,a'')CI(\phi\tau,a) \leq AI(\phi\tau,a')$; in short,

$$abs(CI(\phi)) \leq AI(\phi).$$

Remark. See 10.1.3 for a strengthening to equality (for certain types).
Proof. Follows directly from lemma 6.7. []

We now derive $abs \circ CI \leq AI$, the main result on CI, AI and abs.

8.14. Lemma.
Assume that a, a' and a'' form an assignment triple and satisfy moreover

$$(9) \qquad abs(\tau,a'')(a(x)(dom(a''(\alpha_1)),...,dom(a''(\alpha_n)))) \leq a'(x)(cod(a''(\alpha_1)),...,cod(a''(\alpha_n)))$$

for all term variables x with $\mathbf{type}(x) = \tau = \tau[\alpha_1,\ldots,\alpha_n]$.
Then, for any term t, we have

$$(10) \qquad abs(\mathbf{type}(t),a'')(CI(t,a)) \leq AI(t,a').$$

Proof. With simultaneous induction over the complexity of t resp. p, we prove (10) and (11):

$$(11) \qquad abs(\Phi,a'')(\mathbf{f})(CI(p,a)(dom(\mathbf{f}))) \leq AI(p,a')(cod(\mathbf{f})),$$

where p is a polymorphic term with $\mathbf{type}(p) = \Phi \in \Theta^n\to\Theta$, and $\mathbf{f} = \langle f_1,...,f_n\rangle \in Ar(\mathbb{D}_p{}^n)$ are abstractions.
$t = x$: for simplicity we assume $\mathbf{type}(x) = \tau = \tau[\alpha]$. Now (10) becomes $abs(\tau,a'')(a(x)a(\alpha)) \leq a'(x)a'(\alpha)$ and this follows from (9), since a, a' and a'' form an assignment triple, so $dom(a''(\alpha)) = a(\alpha)$, $cod(a''(\alpha)) = a'(\alpha)$.
$t = k$, a term constant: now (10) becomes $abs(\tau,a'')CI(k) \leq AI(k)$ and this is easily checked for all term constants k.
$p = \bot$: (11) follows from the strictness of $abs(\tau,a'')$ (by lemma 8.12 and definition 6.1.(iii)).
p = fix: follows from lemma 8.13.(ii).
p is some other polymorphic constant: straightforward verification.
$t = rs$, $\mathbf{type}(r) = \sigma\to\tau$, $\mathbf{type}(s) = \sigma$: now (10) becomes

(12) $\quad abs(\tau,a'')(CI(r,a)CI(s,a)) \leq AI(r,a')AI(s,a').$

By lemma 8.13.(i), we have

(13) $\quad abs(\tau,a'')(CI(r,a)CI(s,a)) \leq (abs(\sigma\rightarrow\tau,a'')CI(r,a))(abs(\sigma,a'')CI(s,a))$

and by induction hypothesis (10) for r and s, we have

(14) $\quad abs(\sigma\rightarrow\tau,a'')CI(r,a) \leq AI(r,a'),\ abs(\sigma,a'')CI(s,a) \leq AI(s,a');$

now (13) and (14) yield (12).
$t = \lambda x{:}\rho.s$, $\tau = \rho\rightarrow\sigma$, **type**(x) = $\rho$, **type**(s) = $\sigma$: now (10) becomes

$$(abs(\rho,a'')\twoheadrightarrow abs(\sigma,a''))(\lambda d.CI(s,a[x\rightarrow d^X])) \leq \lambda e.AI(s,a'[x\rightarrow e^X]),$$

i.e.

$$\lambda e.\bigvee\{abs(\sigma,a'')(CI(s,a[x\rightarrow d^X])) \mid abs(\rho,a'')d \leq e\} \leq \lambda e.AI(s,a'[x\rightarrow e^X]),$$

i.e.

(15) $\quad \forall de(abs(\rho,a'')d \leq e \Rightarrow abs(\sigma,a'')(CI(s,a[x\rightarrow d^X])) \leq AI(s,a'[x\rightarrow e^X]))$

and this holds, since if $abs(\rho,a'')d \leq e$, then the assignments $a[x\rightarrow d^X]$, $a'[x\rightarrow e^X]$ and a'' satisfy (9) and we get the conclusion of (15) by induction hypothesis (10) for s.
$t = p\sigma_1...\sigma_n$, **type**(p) = $\delta\alpha_1...\alpha_n.\rho$, $\tau = \rho[\alpha_1{:=}\sigma_1,\ldots,\alpha_n{:=}\sigma_n]$: for simplicity, we assume n=1.
Now (10) becomes

$$abs(\rho[\alpha{:=}\sigma],a'')(CI(p\sigma,a)) \leq AI(p\sigma,a')$$

i.e.

$$(abs(\delta\alpha.\rho,a'')(abs(\sigma,a')))(CI(p,a)(CI(\sigma,a))) \leq AI(p,a')(AI(\sigma,a'))$$

and this follows from induction hypothesis (11) for p, since $dom(abs(\sigma,a'')) = CI(\sigma,a)$, $cod(abs(\sigma,a'')) = AI(\sigma,a)$, by lemma 8.12.
$p = \Lambda\alpha_1...\alpha_n.t$, **type**(t) = $\sigma$, $\tau = \delta\alpha_1...\alpha_n.\sigma$: for simplicity, we assume n=1. Now (11) becomes

$$abs(\delta\alpha.\sigma,a'')(f)(CI(\Lambda\alpha.t,a)(dom(f))) \leq AI(\Lambda\alpha.t,a')(cod(f)) \text{ for abstractions f,}$$

i.e.

$$abs(\sigma,a''[\alpha\rightarrow f])(CI(t,a[\alpha\rightarrow dom(f)])) \leq AI(t,a'[\alpha\rightarrow cod(f)]) \text{ for abstractions f,}$$

and this follows from induction hypothesis (10) for t, since $a[\alpha\rightarrow dom(f)]$, $a'[\alpha\rightarrow cod(f)]$ and $a''[\alpha\rightarrow f]$ form an assignment triple satisfying (9). []

## § 9. STRICTNESS.

With CI and AI at hand, it is not difficult to define strictness and demonstrable strictness.

9.1. Definition.
Let t be a closed term of type $\sigma\to\tau$.
i) t is called *strict* if $CI(t)\bot = \bot$.
ii) t is called *d(emonstrably)-strict* if $AI(t)\bot = \bot$.

Using abs, we can characterise the relation between strict and demonstrably strict:

9.2. Theorem.
Let t be a closed term of type $\sigma\to\tau$. Then:

$$t \text{ is d-strict} \Rightarrow t \text{ is strict.}$$

Proof. We have

$$\begin{aligned}
t \text{ d-strict} &\Leftrightarrow AI(t)\bot = \bot \\
&\Rightarrow abs(\sigma\to\tau)(CI(t))\bot = \bot \qquad (abs\circ CI \le AI, \text{ lemma } 8.14) \\
&\Leftrightarrow abs(\sigma\to\tau)(CI(t))(abs(\sigma)(\bot)) = \bot \\
&\qquad\qquad (abs(\sigma)(\bot) = \bot, \text{ by lemma } 8.12 \text{ and def. } 6.1.(iii)) \\
&\Rightarrow abs(\sigma\to\tau)(CI(t)\bot) = \bot \quad (abs(xy) \le abs(x)abs(y), \text{ lemma } 8.13.(i)) \\
&\Leftrightarrow CI(t)\bot = \bot \\
&\Leftrightarrow t \text{ is strict.}
\end{aligned}$$

□

## § 10. CONCLUDING REMARKS.

We have seen that strictness analysis for a functional language with non-trivial strength can be done using abstract semantics, yielding an approximation of the notion *strict* by *demonstrably strict.* The method given heavily relies on notions and constructions from category theory. It is, in fact, an elaborated example of the use of abstract semantics.

### 10.1. Variants of the method.

We shortly discuss some alternative developments of the method given in this paper.

10.1.1. It is possible to replace all domains by algebraic complete lattices (acl's). This is done as follows:

add $\top$ to N and B;
replace +, in0, in1, case by $+_c$, $in0_c$, $in1_c$, $case_c$;
replace ++ by $+_c$.

The reason we did not choose this option from the beginning is that, contrary to $\bot$ (undefined), $\top$ has no proper meaning in N and in B, at least not in the context of functional languages,where overdefinedness (a natural candidate for the meaning of $\top$) does not play a role.

10.1.2. A less radical variant is to leave N and B as they are, but to replace +, in0, in1 and case by their complete counterparts, observing that $\mathbb{D}$ is closed under these operations, and to replace ++ by $+_c$.

10.1.3. We present an interesting consequence of 10.1.2, viz. strengthening 8.13.(ii) to

$$abs((\tau\to\tau)\to\tau)CI(\phi\tau) = AI(\phi\tau)$$

for types $\tau$ of the fragment $POLYREC^+$ (defined below). This is done as follows.

Definition.
A function f is a *strong abstraction* if f is an abstraction with a continuous right inverse f*, so $f\circ f^* = id_{cod(f)}$. We associate to every strong abstraction f a right inverse f*, and call <f,f*> a *strong abstraction pair.*

Strong abstractions yield a stronger version of 6.7.(ii):

Lemma.
Let f be a strong abstraction, with right inverse f*. Then

$$((f \twoheadrightarrow f) \twoheadrightarrow f)(\mathrm{fix}_{\mathrm{dom}(f)}) = \mathrm{fix}_{\mathrm{cod}(f)}.$$

Proof. We have

$$\begin{aligned}
((f \twoheadrightarrow f) \twoheadrightarrow f)(\mathrm{fix})a &= \bigvee\{f(\mathrm{fix}(d)) \mid (f \twoheadrightarrow f)d \le a\} \\
&= \bigvee\{f(\bigvee\{d^n\bot \mid n\in\omega\}) \mid (f \twoheadrightarrow f)d \le a\} \\
&= \bigvee\{\bigvee\{f(d^n\bot) \mid n\in\omega\}) \mid (f \twoheadrightarrow f)d \le a\} \\
&= \bigvee\{f(d^n\bot) \mid n\in\omega, (f \twoheadrightarrow f)d \le a\} \\
&\ge \bigvee\{f(((f \twoheadrightarrow f)^*a)^n\bot) \mid n\in\omega\}, \text{ for } , (f \twoheadrightarrow f)((f \twoheadrightarrow f)^*a) = a \\
&= \bigvee\{f((\lambda y.f^*(a(fy)))^n\bot) \mid n\in\omega\} \\
&= \bigvee\{f((\lambda y.f^*(a(fy)))^n\bot) \mid n\in\omega\} \\
&= \bigvee\{a^n(f\bot) \mid n\in\omega\} \\
&\ge \bigvee\{a^n\bot \mid n\in\omega\} = \mathrm{fix}(a)
\end{aligned}$$

and the result follows with lemma 6.7.(ii). []

Lemma.
$\mathrm{abs}_N$ and $\mathrm{abs}_B$ have right inverses, and $\times$, $+_c$ and $\twoheadrightarrow$ preserve strong abstractions: the right inverses are obtained by

$$\begin{aligned}
(f\times g)^* &:= (f^*\times g^*) \\
(f+_c g)^* &:= (f^*+_c g^*) \\
(f\twoheadrightarrow g)^* &:= \lambda xy.g^*(x(fy))
\end{aligned}$$

Proof. We only show the correctness of the last definition:

$$\begin{aligned}
&(f\twoheadrightarrow g)((f\twoheadrightarrow g)^*x)y \\
&\quad = (f\twoheadrightarrow g)(\lambda y.g^*(x(fy)))y \\
&\quad = \bigvee\{g(g^*(x(fz))) \mid fz\le y\} \\
&\quad = \bigvee\{x(fz) \mid fz\le y\} \\
&\quad = xy, \text{ for } f(f^*y) = y, \text{ so } \forall y\exists z(fz=y).
\end{aligned}$$

Definition.
i) $(\langle\phi,\phi'\rangle,\langle\psi,\psi'\rangle,\langle f,f^*\rangle,\langle g,g^*\rangle)$ is a *strong abstraction diagram* if $(\langle\phi,\phi'\rangle,\langle\psi,\psi'\rangle,f,g)$ is an abstraction diagram, $\langle f,f^*\rangle$ and $\langle g,g^*\rangle$ are strong abstraction pairs and

$$\begin{aligned}
\phi'\circ f &= g\circ\phi, \\
\phi\circ f^* &= g^*\circ\phi'.
\end{aligned}$$

ii) A *strong abstraction transformation* is defined as an abstraction transformation, but reading strong abstraction for abstraction; we write $H : F \Rightarrow_s G$ if $H$ is a strong abstraction transformation from $F$ to $G$.

We have the following variant of lemma 6.11:

Lemma.
i) $\times : \times \Rightarrow \times$.
ii) $+_c : +_c \Rightarrow +_c$.
iii) If $\langle h,h^* \rangle$ is a strong abstraction pair, then $h \rightarrowtail \cdot : \mathrm{dom}(h) \to \cdot \Rightarrow_s \mathrm{cod}(h) \to \cdot$. Here $h \rightarrowtail \cdot$ abbreviates $\lambda x.\ h \rightarrowtail x$, etc.
iv) Strong abstraction pairs are closed under composition.
Proof. (i),(ii): easy.
(iii) Assume that $\langle h,h^* \rangle$ is a strong abstraction pair and $(\langle \phi,\phi' \rangle,\langle \psi,\psi' \rangle,\langle f,f^* \rangle,\langle g,g^* \rangle)$ is a strong abstraction diagram. We have to show that $(\langle \phi \circ \cdot,\phi' \circ \cdot \rangle,\langle \psi \circ \cdot,\psi' \circ \cdot \rangle,$ $\langle h \rightarrowtail f,(h \rightarrowtail f)^* \rangle,\langle h \rightarrowtail g,(h \rightarrowtail g)^* \rangle)$ is a strong abstraction diagram, i.e.

(1) $\quad (\phi' \circ \cdot) \circ (h \rightarrowtail f) = (h \rightarrowtail g) \circ (\phi \circ \cdot)$,
(2) $\quad (\phi \circ \cdot) \circ (h \rightarrowtail f)^* = (h \rightarrowtail g)^* \circ (\phi' \circ \cdot)$.

(1): we have

$$((\phi' \circ \cdot) \circ (h \rightarrowtail f))(x)y = \phi'(\bigvee\{f(xz) \mid hz \leq y\}) = \bigvee\{\phi'(f(xz)) \mid hz \leq y\})$$
$$((h \rightarrowtail g) \circ (\phi \circ \cdot))(x)y = \bigvee\{g(\phi(xz)) \mid hz \leq y\})$$

and these are equal because of $\phi' \circ f = g \circ \phi$.
(2): here

$$((\phi \circ \cdot) \circ (h \rightarrowtail f)^*)(x)y = \phi(f^*(x(hy))),$$
$$((h \rightarrowtail g)^* \circ (\phi' \circ \cdot))(x)y = g^*(\phi'(x(hy))),$$

and their equality follows from $\phi \circ f^* = g^* \circ \phi'$. []

We turn to chains. If $\langle f_n \rangle_n$ and $\langle f^*_n \rangle_n$ are chains with $\mathrm{dom}(f_n) = \mathrm{cod}(f^*_n) = X_n$, $\mathrm{cod}(f_n) = \mathrm{dom}(f^*_n) = Y_n$ and $\forall n(f_n \circ f^*_n = 1_{Y_n})$, then (by 7.12.(vi)) we have $(\lim_n f_n) \circ (\lim_n f^*_n) = \lim_n (f_n \circ f^*_n) = 1_Y$, where $Y = \lim_n Y_n$. So for limits of abstractions we have

$$(\lim_n f_n)^* = \lim_n (f^*_n).$$

This leads to the following variants of 7.19.(ii) and 7.21:

if H is a strong abstraction transformation, then Fix(H) is a strong abstraction;
if $H : F \Rightarrow_S G$, then $Fix(H) : Fix(F) \Rightarrow_S Fix(G)$.

Now we consider the *positive fragment* $POLYREC^+$ of POLYREC, where types of the form $\sigma \to \tau$ are only allowed if $\sigma$ is closed, i.e. contains no free type variables. For types $\tau$ and type functions $\Phi$ of $POLYREC^+$ we can strengthen lemma 8.12 and 8.13.(ii) to:

$abs(\tau) \in [CI(\tau) \to AI(\tau)]$ is a strong abstraction,
$abs(\Phi) : CI(\Phi) \Rightarrow_S AI(\Phi)$;
$abs((\tau \to \tau) \to \tau)CI(\phi\tau) = AI(\phi\tau)$.

10.3. Perspectives for future research.
The following items, not covered in this paper, seem of interest for future research.
i) Extension of the method to stronger languages, e.g. second-order lambda calculus (see 2.4), or the extension $ML^+$ of ML defined in [My84] (see also [KTU88]). The obvious question is: given some concrete interpretation, find an abstract interpretation and an abstraction mapping between the two.
ii) It seems promising to study the behaviour of POLYREC as a term rewrite system. A first topic is an analysis of the relation between strict and needed, as mentioned in 1.1. Furthermore it might be interesting to try and develop *abstract* rewrite rules to compute demonstrable strictness.
iii) Refinement of the method, e.g. by considering finite approximations of limits of chains, in order to make computation of demonstrable strictness feasible.
iv) Analysis concerning the algorithmic complexity of the method and its variants.

## APPENDIX.

In this appendix we discuss the definition of abstraction diagram (6.8.(i)). It contains the commutation condition $\phi' \circ f = g \circ \phi$. Why this condition? There are alternative conditions: together there are four possibilities, viz.

(1) $\phi' \circ f \circ \psi = g$,
(2) $\phi' \circ f = g \circ \phi$,
(3) $f \circ \psi = \psi' \circ g$,
(4) $f = \psi' \circ g \circ \phi$;

we consider them (and some weakenings) in the following lemma.

Lemma.
Let (p,p',f,g) be an abstraction diagram as defined in 6.8.(i), but without the condition $\phi' \circ f = g \circ \phi$ (i.e.(2)). Then
i) (1) $\Rightarrow$ (2) $\Rightarrow$ (4);
ii) (1) $\Rightarrow$ (3) $\Rightarrow$ (4);
iii) $\phi' \circ f \circ \psi \leq g \Leftrightarrow \phi' \circ f \leq g \circ \phi \Leftrightarrow f \circ \psi \leq \psi' \circ g \Leftrightarrow f \leq \psi' \circ g \circ \phi$.
Proof. Simple application of the properties of projections. □

Before we discuss alternative definitions of abstraction diagram, we take a look at the place they are used: lemma 7.18.(i) on chains of abstraction transfomations. The typical situation is the beginning of such a chain, i.e. the diagram determined by the domains $\underline{1}$, $\underline{1}$, $F(\underline{1})$, $G(\underline{1})$, the projections $\langle \phi, \psi \rangle$, $\langle \phi', \psi' \rangle$ and the abstractions $1_{\underline{1}}$, $H(\underline{1})$. Now it is easy to see that (2) (and hence (4)) always holds for such a diagram (for $\underline{1}$ is a terminal object), and (3) holds if $\psi$, $\psi'$ and $H(\underline{1})$ are strict. Condition (1), however, is rarely met: it requires $H(\underline{1})$ to be constant (with $\psi'(\bot)$ as value). Therefore we shall discard condition (1) in the rest of our considerations.

It turns out that (2) is the only condition with makes $\twoheadrightarrow : \rightarrow \Rightarrow \rightarrow$ (lemma 6.11.(iii)) true. We show this by giving a counterexample to $\twoheadrightarrow : \rightarrow \Rightarrow \rightarrow$ for abstraction diagrams based on (3) or (4). It consists of three abstraction diagrams satisfying (3) (hence (4)) and a demonstration that the resulting diagram does not satisfy (4) (hence neither (3)).
Let

$\underline{1} := \{\bot\}$,
$\underline{2} := \{\bot, \top\}$ with $\bot \leq \top$
$B := \{\bot, 0, 1\}$ with $\bot \leq 0, 1$
$f : B \rightarrow \underline{2}$ with $f(\bot) = f(0) = \bot$, $f(1) = \top$

$\phi : \underline{2} \to B$ with $\phi(\bot) = \bot$, $\phi(\top) = 0$

$\psi : B \to \underline{2}$ with $\psi(\bot) = \bot$, $\psi(0) = \psi(1) = \top$

then (I) = $(<\phi,\psi>,<1_{\underline{2}},1_{\underline{2}}>,1_{\underline{2}},f)$, (II) = $(<1_B,1_B>,<1_{\underline{2}},1_{\underline{2}}>,f,f)$ and (III) = $(<1_{\underline{2}},1_{\underline{2}}>,<1_{\underline{2}},1_{\underline{2}}>,1_{\underline{2}},1_{\underline{2}})$ are abstraction diagrams satisfying (3). In (II) and (III) even (1) holds, but not in (III), for

$$1_{\underline{2}} \circ 1_{\underline{2}} = 1_{\underline{2}} \neq \lambda x.\bot = f \circ \phi.$$

Now we shall show that in the diagram ((I) $\to$ (II)) $\to$ (III) we have

(*) $\quad f^* \neq \psi'^* \circ g^* \circ \phi^*,$

where

$f^* := ((1_{\underline{2}} \twoheadrightarrow f) \twoheadrightarrow 1_{\underline{2}})$,
so $f^*(x)(y) = \bigvee[xz \mid z \in [\underline{2} \to B] \wedge \forall u \in \underline{2}\ (f(zu) \leq yu)\}$;

$g^* := ((f \twoheadrightarrow f) \twoheadrightarrow 1_{\underline{2}})$,
so $g^*(x)(y) = \bigvee[xz \mid z \in [B \to B] \wedge \forall b \in B\ (f(zb) \leq y(fb))\}$;

$\phi^* := ((\phi \to 1_B) \to 1_{\underline{2}})$,
so $\phi^*(x)(y) = x(y \circ \phi)$;

$\psi'^* := ((1_{\underline{2}} \to 1_B) \to 1_{\underline{2}})$,
so $\psi'^*(x)(y) = xy$.

We do this by showing

$$(f^*)(x)y = \bot \neq \top = (\psi'^* \circ g^* \circ \phi^*)(x)y,$$

where

$x \in [[\underline{2} \to B] \to \underline{2}]$, defined by $x(z)=\top$ if $z\top=1$, $x(z)=\bot$ otherwise;
$y \in [\underline{2} \to \underline{2}]$, defined by $y = 1_{\underline{2}}$.

We have

$f^*(x)y = \bigvee[xz \mid z \in [\underline{2} \to B] \wedge \forall u \in \underline{2}\ (f(zu) \leq yu)\}$
$= \bigvee[\top \mid \exists z \in [\underline{2} \to B](z\top=1 \wedge f(z\bot) \leq \bot \wedge f(z\top) \leq \top))\}$
$= \top$ (take $z$ with $z\bot=\bot$, $z\top=1$)

and

$$
\begin{aligned}
(\psi'^* \circ g^* \circ \phi^*)(x)y &= \psi'^*(g^*(\phi^* x))y \\
&= g^*(\phi^* x)y \\
&= \bigvee\{(\phi^* x)z \mid z \in [B \to B] \wedge \forall b \in B(f(zb) \leq y(fb))\} \\
&= \bigvee\{x(z \circ \phi) \mid z \in [B \to B] \wedge \forall b \in B(f(zb) \leq fb))\} \\
&= \bigvee\{\top \mid \exists z \in [B \to B](z(\phi\top) = 1 \wedge f(z\bot) \leq f\bot \wedge f(z0) \leq f0 \wedge f(z1) \leq f1)\} \\
&= \bigvee\{\top \mid \exists z \in [B \to B](z0 = 1 \wedge f(z\bot) = \bot \wedge f1 = \bot)\} \\
&= \bot, \text{ for } f1 = \top \neq \bot.
\end{aligned}
$$

**REFERENCES.**


[Ba84] H.P. Barendregt, *The Lambda Calculus, its syntax and semantics*, North-Holland Publishing Company, Amsterdam etc., 1984.

[BKKS86] H.P. Barendregt, J.R. Kennaway, J.W. Klop, M.R. Sleep, *Needed reduction and spine strategies for the lambda calculus*, Report CS-R8621, Centre for Mathematics and Computer Science, 1986.

[BHA86] G.L. Burn, C.L. Hankin, S. Abramsky, *Strictness analysis for higher order functions*, Science of Computer Programming 7 (1986) 249 - 278.

[Er75] Iu.L. Ershov, *Theorie der Numerierungen II*, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik 21 (1975) 473 - 584.

[GL84] P. Giannini, G. Longo, *Effectively given domains and lambda-calculus models*, Information and Control 62 (1984) 36 - 63.

[KTU88] A.J. Kfoury, J. Tiuryn, P. Urzyczyn, *A proper extension of ML with an effective type assignment*, in: *Proc. 15$^{th}$ ACM Symposium on Principles of Programming Languages*, 1988 (to appear).

[Lo79] G. Longo, *Ricorsività nei tipi superiori: un'introduzione alle caratterizzazioni di Ershov ed Hyland*, Note Scientifiche S - 79 - 9, Università degli Studi di Pisa, Istituto di Scienze dell'Informazione (1979).

[My84] A. Mycroft, *Polymorphic type schemes and recursive definition*, in: M. Paul and B. Robinet (eds.), *International Symposium on Programming*, LNCS 167, Springer-Verlag, 1984.

[Sc76] D.S. Scott, *Data types as lattices*, SIAM Journal of Computing 5 (1976) 522 - 587.

[SP82] M. Smyth, G. Plotkin, *The category-theoretic solution of recursive domain equations*, SIAM Journal of Computing 11 (1982) 761-783.

**Logic Group Preprint Series**
Department of Philosophy
University of Utrecht
Heidelberglaan 2
3584 CS Utrecht
The Netherlands

nr. 1 C.P.J. Koymans, J.L.M. Vrancken, *Extending Process Algebra with the empty process*, September 1985.
nr. 2 J.A. Bergstra, *A process creation mechanism in Process Algebra*, September 1985.
nr. 3 J.A. Bergstra, *Put and get, primitives for synchronous unreliable message passing*, October 1985.
nr. 4 A.Visser, *Evaluation, provably deductive equivalence in Heyting's arithmetic of substitution instances of propositional formulas*, November 1985.
nr. 5 G.R. Renardel de Lavalette, *Interpolation in a fragment of intuitionistic propositional logic*, January 1986.
nr. 6 C.P.J. Koymans, J.C. Mulder, *A modular approach to protocol verification using Process Algebra* , April 1986.
nr. 7 D. van Dalen, F.J. de Vries, *Intuitionistic free abelian groups*, April 1986.
nr.8 F. Voorbraak, *A simplification of the completeness proofs for Guaspari and Solovay's R*, May 1986.
nr.9 H.B.M. Jonkers, C.P.J. Koymans & G.R. Renardel de Lavalette, *A semantic framework for the COLD-family of languages*, May 1986.
nr.10 G.R. Renardel de Lavalette, *Strictheidsanalyse*, May 1986.
nr.11 A. Visser, *Kunnen wij elke machine verslaan? Beschouwingen rondom Lucas' argument*, July 1986.
nr.12 E.C.W. Krabbe, *Naess's dichotomy of tenability and relevance*, June 1986.
nr.13 Hans van Ditmarsch, *Abstractie in wiskunde, expertsystemen en argumentatie*, Augustus 1986
nr.14 A.Visser, *Peano's Smart Children, a provability logical study of systems with built-in consistency* , October 1986.
nr.15 G.R.Renardel de Lavalette, *Interpolation in natural fragments of intuitionistic propositional logic*, October 1986.
nr.16 J.A. Bergstra, *Module Algebra for relational specifications*, November 1986.
nr.17 F.P.J.M. Voorbraak, *Tensed Intuitionistic Logic*, January 1987.
nr.18 J.A. Bergstra, J. Tiuryn, *Process Algebra semantics for queues*, January 1987.
nr.19 F.J. de Vries, *A functional program for the fast Fourier transform*, March 1987.
nr.20 A. Visser, *A course in bimodal provability logic*, May 1987.
nr.21 F.P.J.M. Voorbraak, *The logic of actual obligation, an alternative approach to deontic logic*, May 1987.
nr.22 E.C.W. Krabbe, *Creative reasoning in formal discussion*, June 1987.
nr.23 F.J. de Vries, *A functional program for Gaussian elimination*, September 1987.
nr.24 G.R. Renardel de Lavalette, *Interpolation in fragments of intuitionistic propositional logic*, October 1987.(revised version of no. 15)
nr.25 F.J. de Vries, *Applications of constructive logic to sheaf constructions in toposes*, October 1987.
nr.26 F.P.J.M. Voorbraak, *Redeneren met onzekerheid in expertsystemen*, November 1987.
nr.27 P.H. Rodenburg, D.J. Hoekzema, *Specification of the fast Fourier transform algorithm as a term rewriting system*, December 1987.

nr.28 D. van Dalen, *The war of the frogs and the mice, or the crisis of the Mathematische Annalen,* December 1987.

nr.29 A. Visser, *Preliminary Notes on Interpretability Logic,* January 1988.

nr.30 D.J. Hoekzema, P.H. Rodenburg, *Gauß elimination as a term rewriting system*, January 1988.

nr. 31 C. Smoryński, *Hilbert's Programme,* January 1988.

nr. 32 G.R. Renardel de Lavalette, *Modularisation, Parameterisation, Interpolation,* January 1988.

nr. 33 G.R. Renardel de Lavalette, *Strictness analysis for POLYREC, a language with polymorphic and recursive types*, March 1988.