A NOTE ON POWERDOMAINS AND MODALITY

Glynn Winskel
Computer Science Department
Carnegie-Mellon University
Pittsburgh, Pa.

September 1983

A version of this paper has been published in a volume of the Springer-Verlag Lecture Note Series containing the proceedings of the conference "Foundations of Computation Theory" held in Sweden, August 1983. A version has been submitted to the journal "Theoretical Computer Science".

The research reported in this paper was supported in part by funds from the Computer Science Department of Carnegie-Mellon University, and by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 3597, monitored by the Air Force Avionics Laboratory under Contract F33615-81-K-1539. The views and conclusions contained in it are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US Government.

A NOTE ON POWERDOMAINS AND MODALITY

by

Glynn Winskel
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

This note shows a simple connection between powerdomains and modal assertions that can be made about nondeterministic computations. We consider three kinds of powerdomains, the Plotkin powerdomain, the Smyth powerdomain and one Christened the Hoare powerdomain by Plotkin because it captures the partial correctness of a nondeterministic program. The modal operators are $\Box$ for "inevitably" and $\Diamond$ for "possibly". It is shown in a precise sense how the Smyth powerdomain is built up from assertions about the inevitable behaviour of a process, the Hoare powerdomain is built up from assertions about the possible behaviour of a process while Plotkin powerdomain is built up from from both kinds of assertions taken together.

0. Introduction.

Powerdomains are the complete partial orders (c.p.o.'s, which here we call domains) in which to denote nondeterministic computations. They can be regarded as the domain analogues of powerset with elements which represent the "sets" of different courses a nondeterministic computation can follow. Simple forms of powerdomains were first introduced independently by Egli and Milner but their constructions only worked correctly for powerdomains of flat , or discrete, domains. A breakthrough was made by Gordon Plotkin when he gave a construction of a powerdomain for a rather general class of domains. His construction works well for those domains which are algebraic *i.e.* domains which have a basis of isolated or finite elements [P1]. Although the Plotkin powerdomain construction produces an algebraic domain from an algebraic domain it is not the case that algebraic domains in general are closed under exponentiation, or function space, a vital construction in denotational semantics. However Plotkin was able to show how by restricting constructions to a slightly smaller class of domains, those algebraic domains which are SFP, one obtained a category closed simultaneously under his powerdomain construction and function space. His powerdomain construction could be used in the recursive definition of domains, justifying the recursive definition of domains like that of resumptions which have been used to give denotational semantics to some parallel programming languages in which atomic actions are interleaved—see [P1] and [HP] for details.

The original presentation of powerdomains in [P1] was hard to follow and the construction was streamlined by Mike Smyth in [Smy] which also introduced a new powerdomain subsequently called the Smyth powerdomain. The Smyth powerdomain identifies more processes than Plotkin's when one uses it to give a denotational semantics. However it has the technical advantage that the very pleasant category of consistently complete algebraic domains are closed under the Smyth powerdomain construction. The consistently complete algebraic domains are those represented in [S]; they can be thought of algebraic lattices with isolated top–elements removed. The work of Dana Scott has shown the nice properties of this category. In particular it is closed under function space. Unfortunately it is not closed under the Plotkin powerdomain.

To secure the work that follows we present the main results on powerdomains. The reader should have few problems filling in the proofs. For the full details the reader can refer to [Smy], though to be honest, here in the introduction we follow the lines suggested by Smyth—and earlier by Plotkin in [P1]—rather than the lines Smyth actually follows, using generating trees and finitely generable sets. (We use generating trees in the next section.)

Recall a *directed set* of a partial order $(D, \sqsubseteq)$ is a non–null subset $S \subseteq D$ such that $\forall s, t \in S \exists u \in S.\, s \sqsubseteq u \ \&\ t \sqsubseteq u$. A *complete partial order*, a c.p.o., is a partial order $(D, \sqsubseteq)$ which has a least element $\bot$ and all least upper bounds of directed subsets. An *isolated (or finite)* element of a c.p.o. $(D, \sqsubseteq)$ is an element $x \in D$ such that for any directed subset $S \subseteq D$ when $x \sqsubseteq \bigsqcup S$ there is $s \in S$ such that $x \sqsubseteq s$. We write D^0 for the set of isolated elements of D. Intuitively the isolated elements are that information which a computation can realise in finite time., When there are enough isolated elements to form a basis the c.p.o. is said to be *algebraic i.e.* a c.p.o. $(D, \sqsubseteq)$ is *algebraic* iff for all $x \in D$ we have $x = \bigsqcup\{ e \sqsubseteq x \mid e \in D^0 \}$. When $(D, \sqsubseteq)$ is algebraic and D^0 is countable, D is said to be *countably algebraic* or simply ω–algebraic.

Smyth showed easy constructions of the powerdomains of ω–algebraic domains D. They were built from finite, non–null sets of isolated elements of D, which we call $M[D]$. Given an ordered set there are three natural ways to pre–order its subsets. Let us list the pre–orders: For A, B in $M[D]$, write

$$\begin{aligned} A \preceq_0 B &\Leftrightarrow \forall b \in B \exists a \in A.\, a \sqsubseteq b \\ A \preceq_1 B &\Leftrightarrow \forall a \in A \exists b \in B.\, a \sqsubseteq b \\ A \preceq_2 B &\Leftrightarrow A \preceq_0 B \ \&\ A \preceq_1 B. \end{aligned}$$

Smyth only considered two, the preorder $\preceq_2$, called the *Egli–Milner ordering*, and the preorder $\preceq_0$ from which he obtained the Plotkin and Smyth powerdomains respectively. However a similar treatment also yields a powerdomain associated with the pre–order $\preceq_1$. This powerdomain has been called the Hoare powerdomain by Plotkin because of its relation with C. A. R. Hoare's work on partial correctness. To be fair this powerdomain has been invented independently by many people, Plotkin and David Park to name two while the "Traces model" for a version of CSP [Ho] provides an instance of this powerdomain. The Hoare powerdomain ignores the divergence of computations completely but like the Smyth powerdomain works smoothly within the category of consistently complete algebraic domains.

There is a standard way to get an algebraic domain from a preorder with a least element, often called *completion by ideals*—see *e.g.* Scott's work, and [Gue]. The method is to take ideals(=left–closed, directed subsets) of the preorder and order them by inclusion to obtain an algebraic domain with isolated elements which correspond to equivalence classes of elements of the preorder under the preorder's natural equivalence. More formally,

0.1 Proposition. *Let $(P, \preceq)$ be a preorder with a least element $\perp \preceq p$ for all p in P. Define the ideals $\mathcal{I}(P)$ by*

$$X \in \mathcal{I}(P) \Leftrightarrow \emptyset \neq X \subseteq P \ \& \forall p, q.\, q \preceq p \in X \Rightarrow q \in X \ \& \\ \forall p, q \in X \exists r \in X.\, p \preceq r \ \& \ q \preceq r.$$

Then $(\mathcal{I}(D), \subseteq)$ is an algebraic domain, with isolated elements $\{ q \in P \mid q \preceq p \}$ for $p \in P$.

The three different powerdomains of D are obtained by completing by ideals the three preorders $\preceq_0$, $\preceq_1$, $\preceq_2$ on $M[D]$. As one would expect they carry natural notions of union induced by the union operation on the sets $M[D]$.

0.2 Proposition.

The Smyth powerdomain of D, written $P_0[D]$, is the completion by ideals of the preorder $(M[D], \preceq_0)$ i.e. $(P_0[D], \sqsubseteq_0) = (\mathcal{I}(M[D], \preceq_0), \subseteq)$; it carries a natural continuous union operation

$$X \cup_0 Y =_{def} \{ A \cup B \mid A \in X \ \& \ B \in Y \}$$

which is associative, commutative, idempotent (i.e. $X \cup_0 X = X$) and satisfies the law $X \cup_0 Y \sqsubseteq X$.

The Hoare powerdomain of D, written $P_1[D]$, is the completion by ideals of the preorder $(M[D], \preceq_1)$ i.e. $(P_1[D], \sqsubseteq_1) = (\mathcal{I}(M[D], \preceq_1), \subseteq)$; it carries a natural continuous union operation

$$X \cup_1 Y =_{def} \{ A \cup B \mid A \in X \ \& \ B \in Y \}$$

which is associative, commutative, idempotent and satisfies the law $X \sqsubseteq X \cup_0 Y$.

The Plotkin powerdomain of D, written $P_2[D]$, is the completion by ideals of the preorder $(M[D], \preceq_2)$ i.e. $(P_2[D], \sqsubseteq_2) = (\mathcal{I}(M[D], \preceq_2), \subseteq)$; it carries a natural continuous union operation

$$X \cup_2 Y =_{def} \{ A \cup B \mid A \in X \ \& \ B \in Y \}$$

which is associative, commutative and idempotent.

It is easily verified that the union operations are well defined. For the Hoare powerdomain there is an even simpler construction. The Hoare powerdomain of an algebraic domain is isomorphic to the non-null left-closed subsets of its isolated elements, ordered by inclusion.

0.3 Proposition. *Let $\mathcal{L}(D^0)$ consist of the non-null, left-closed subsets of D^0 i.e. $X \in \mathcal{L}(D^0) \Leftrightarrow \emptyset \neq X \subseteq D^0 \ \& \ \forall d, e.\, d \sqsubseteq e \in X \Rightarrow d \in X$. Then $\mathcal{L}(D^0) \cong P_1[D]$, the Hoare powerdomain.*

Thus powerdomains of algebraic domains have a very simple construction. They can be looked on as kinds of algebras with a binary union operation over a domain as carrier. This approach led to a very pleasing characterisation of powerdomains due to Matthew Hennessy and Gordon Plotkin which in fact establishes the existence of powerdomains of arbitrary c.p.o.'s—see [HP]. Define a nondeterministic algebra to be a domain with a continuous binary "union" operation which is associative, commutative and idempotent. Define a homomorphism of such algebras to be a continuous function which preserves the "union" operation. The Plotkin powerdomain is the free algebra with respect to the obvious forgetful functor from algebras to domains. Similarly the Smyth powerdomain is the free algebra amongst those algebras which in addition satisfy $X \cup Y \sqsubseteq X$ and Hoare powerdomain is the free algebra amongst those satisfying $X \sqsubseteq X \cup Y$.

This completes our summary of the properties of powerdomains. In the following we show another way to view powerdomains as consisting of sets of modal assertions that can be made about nondeterministic computations. This indicates a relation between the denotational semantics of nondeterminism and work in the temporal logic of programs, for example [LO].

1. Nondeterministic computations.

Throughout let $(D, \sqsubseteq)$ be an ω-algebraic domain with isolated elements D^0.

We imagine a nondeterministic computation which at each state determines an element of D^0. The element could be for example a finite sequence of values which the computation extends as it progresses. Assume that the nondeterminism is bounded so that each state has only a finite number of next states and for simplicity that the states form a tree with respect to the next-state relation.

1.1 Definition. A *nondeterministic D-computation* has the form $(T, \longrightarrow, val)$ where $(T, \longrightarrow)$ is a finitely branching tree and *val* is a map to D^0 such that

$$\forall t, t' \in T. t \longrightarrow t' \Rightarrow val(t) \sqsubseteq val(t').$$

1.2 Notation. Let $(T, \longrightarrow)$ be a tree.

Let $t \in T$. Write $t \to$ for $\exists t' \in T.t \longrightarrow t'$ and $t \not\to$ for $\nexists t' \in T.t \longrightarrow t'$.

A *branch* is a sequence $t_0, t_1, \ldots, t_n, \ldots$ where t_0 is the root node and $t_n \longrightarrow t_{n+1}$ for each $n+1$ at which the sequence is defined. By a *maximal branch* of $(T, \longrightarrow)$ we mean a branch which is either infinite, or finite of the form $t_0, t_1, \ldots, t_n$ with $t_n \not\to$.

In [Smy], Smyth extracts the finitely generable sets from labelled trees like those above; for convenience he requires they satisfy the additional axiom $\forall t \in T \exists t' \in T.t \longrightarrow t'$.

2. The Smyth powerdomain.

We now make a little language to talk about nondeterministic D–computations. The atomic statements are just elements of D^0. To get Smyth's powerdomain we include disjunction "$\vee$" and the "inevitably" modality "$\Box$" in the language.

2.1 Definition. Let the language L_0 be the least set including D^0 and such that

$$s \in L_0 \Rightarrow \Box s \in L_0 \quad \text{and,}$$
$$s, s' \in L_0 \Rightarrow (s \vee s') \in L_0.$$

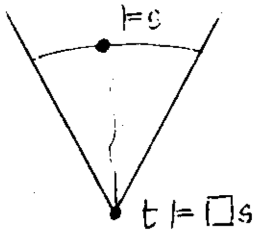
2.2 Definition. (The satisfaction relation)

Let $(T, \longrightarrow, val)$ be a nondeterministic D-computation. Define $\models_T$ to be the least relation included in $T \times L_0$ such that

$$a \sqsubseteq val(t) \Rightarrow t \models_T a \quad \text{for } a \in D^0,$$
$$t \models_T s \text{ or } t \models_T s' \Rightarrow t \models_T (s \vee s') \quad \text{and,}$$
$$t \models_T s \text{ or } (t \to \& \forall t'.t \longrightarrow t' \Rightarrow t' \models_T \Box s) \Rightarrow t \models_T \Box s.$$

Alternatively $\models$ can be constructed inductively as the union of a chain of relations got by starting at the null relation and at each stage growing the relation in accord with the three clauses above. Because T is finitely–branching the closure ordinal of this associated inductive definition will be ω.

Another way to define satisfaction for $\Box$-statements is to say $t \models_T \Box s$ iff every maximal branch in the subtree out of t has a node which satisfies s. In a picture:



This means our satisfaction relation for statements $\Box(s \vee s')$ is the same as that for statements $(s \vee s')$ in Beth trees—see [D].

Suppose we are not interested in the statements which are initially true (at the root node) of a nondeterministic computation but in those statements which are inevitably true. Then it is natural to associate the following set of assertions with a nondeterministic computation.

2.3 Definition. Let $\mathcal{T} = (T, \rightarrow, val)$ be a nondeterministic D-computation with root node t. Write $\models_{\mathcal{T}} s$ for $t \models_{\mathcal{T}} s$. Define

$$V_0(\mathcal{T}) = \{ \Box s \in L_0 \mid \models_{\mathcal{T}} \Box s \}.$$

Such sets of assertions induce an obvious preorder on nondeterministic D-computations.

2.4 Definition. Let $\mathcal{T}$ and $\mathcal{T}'$ be nondeterministic D-computations. Define

$$\mathcal{T} \preceq_0 \mathcal{T}' \Leftrightarrow V_0(\mathcal{T}) \subseteq V_0(\mathcal{T}').$$

Quotienting the preorder $\preceq_0$ on nondeterministic computations by the equivalence $\simeq_0 =_{def} \preceq_0 \cap \preceq_0^{-1}$ we obtain the Smyth powerdomain (see [Smy]).

2.5 Theorem. *Let $\mathbf{T}$ be the class of nondeterministic D-computations. The Smyth powerdomain $P_0[D]$ is isomorphic to the quotient $(\mathbf{T}/\simeq_0, \preceq_0 /\simeq_0)$ and to the order $(\{ V_0(\mathcal{T}) \mid \mathcal{T} \in \mathbf{T} \}, \subseteq)$.*

Proof. (sketched) Write $s \equiv s'$ iff $\forall \mathcal{T}.(\models_{\mathcal{T}} s \Leftrightarrow \models_{\mathcal{T}} s')$. Clearly $s \vee (s' \vee s'') \equiv (s \vee s') \vee s''$ and $s \vee s' \equiv s' \vee s$ so we need not trouble ourselves over the order of disjunctions. It is easy to check that $\Box(\Box s) \equiv \Box s$ and $\Box(s \vee \Box s') \equiv \Box(s \vee s')$. A simple induction shows that each each $\Box s \in L_0$ is $\equiv$-equivalent to a normal form $\Box(a_0 \vee \cdots \vee a_n)$ for some $a_0, \ldots, a_n \in D^0$. Thus the statements $V_0(\mathcal{T})$, satisfied by $\mathcal{T}$, are equivalent to those of a simpler form. Now, to each computation T we associate a subset of $M[D]$ given by $I(T) = \{\{a_0, \ldots, a_n\} \mid \models_T \Box(a_0 \vee \cdots a_n)\}$. From the properties of $\Box$-statements it follows that $I(T)$ is an ideal w.r.t. $\preceq_0$ and so an element of $P_0[D]$. Conversely any element x of $P[D]$ can be obtained as $x = I(T)$ for some computation T constructed as follows : The ideal x is generated by an ω-chain $X_0 \preceq_0 \cdots X_n \preceq_0 \cdots$; inductively construct T so that the labels of its nodes at height n form the set X_n. It follows that there is a 1–1, $\subseteq$-preserving correspondence between sets $V_0(T)$ and the elements of $P_0[D]$. This implies the result. ∎

Remark. Clearly from the above proof, the result holds if we restrict the statements to those of the form $\Box(a_0 \vee \cdots \vee a_n)$ where $a_0, \ldots, a_n \in D^0$.

3. The Hoare powerdomain.

To get another powerdomain, the Hoare powerdomain—sometimes called the upside-down-Smyth powerdomain—we look at assertions built using the modal operator $\Diamond$ standing for "possibly".

3.1 Definition. Define the language L_1 to be the language which is the least set including the atomic statements D^0 such that

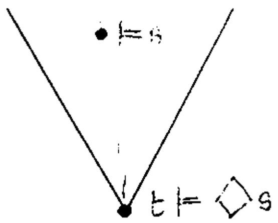
$$s \in L_1 \Rightarrow \Diamond s \in L_1.$$

3.2 Definition. (The satisfaction relation)

Let $(T, \longrightarrow, val)$ be a nondeterministic D-computation. Define $\models_T$ to be the least relation included in $T \times L_1$ such that

$$a \sqsubseteq val(t) \Rightarrow t \models_T a \quad \text{and,}$$
$$t \models_T s \text{ or } \exists t'.t \longrightarrow t' \;\&\; t' \models_T \Diamond s \Rightarrow t \models_T \Diamond s.$$

Alternatively satisfaction for $\Diamond$-statements could be defined by saying $t \models_T \Diamond s$ iff there is a branch out of t which has a node which satisfies s. In a picture:



As before we can preorder nondeterministic computations by the statements they satisfy. This time we are interested in their possibilities.

3.3 Definition. Let $T = (T, \rightarrow, val)$ be a nondeterministic D-computation with root node t. Write $\models_T s$ for $t \models_T s$. Define

$$V_1(T) = \{ \Diamond s \in L_1 \mid \models_T s \}.$$

For nondeterministic D-computations, T and T' define

$$T \preceq_1 T' \Leftrightarrow V_1(T) \subseteq V_1(T').$$

The preorder $\preceq_1$, when quotiented by the equivalence $\simeq_1 =_{def} \preceq_1 \cap \preceq_1^{-1}$, gives a powerdomain associated with the other half of the Egli–Milner ordering to Smyth's—called the upside-down-Smyth powerdomain or Hoare powerdomain.

3.4 Theorem. *Let $\mathcal{T}$ be the class of nondeterministic D-computations. The Hoare powerdomain $P_1[D]$ is isomorphic to the quotient $(\mathcal{T}/\simeq_1, \preceq/\simeq_1)$ and to the order $(\{ V_1(T) \mid T \in \mathcal{T} \}, \subseteq)$.*

Proof. Write $s \equiv s'$ iff $\forall T.(\models_T s \Leftrightarrow \models_T s')$. Each statement $\Diamond s \in L_1$ is $\equiv$-equivalent to a normal form $\Diamond a$ for some $a \in D^0$. For a computation T, define $I(T) = \{ a \in D^0 \mid \models_T \Diamond s \}$. Then $I(T) \in \mathcal{L}(D^0)$, the non-null, left-closed subsets of D^0. Because each statement has a normal form it is clear that $T \preceq_1 T' \Leftrightarrow I(T) \subseteq I(T')$. However I is onto $\mathcal{L}(D^0)$ (Given $x \in \mathcal{L}(D^0)$, the reader is invited to construct T so $I(T) = x$). But $\mathcal{L}(D^0) \cong P_1[D]$, establishing the result. ∎

Remark. The same result would hold if L_1 was restricted to sentences of the form $\Diamond a$ for $a \in D^0$ or expanded to include sentences of the form $\Diamond(s \vee s')$.

4. The Plotkin powerdomain.

Finally the Plotkin powerdomain is obtained by considering information about both the inevitable and possible behaviour of a computation.

4.1 Definition. Let L_2 be the least language containing the elements D^0 as atoms and such that

$$s, s' \in L_2 \Rightarrow (s \vee s') \in L_2,$$
$$s \in L_2 \Rightarrow \Box s \in L_2,$$
$$s \in L_2 \Rightarrow \Diamond s \in L_2.$$

4.2 Definition. (The satisfaction relation.)

Let $(T, \longrightarrow, val)$ be a nondeterministic D–computation. Let $\models_T$ be the least relation included in $T \times L_2$ which satisfies:

$$\begin{aligned}
&a \sqsubseteq val(t) \Rightarrow t \models_T a \quad \text{for } a \in D^0, \\
&t \models_T s \text{ or } t \models_T s' \Rightarrow t \models_T s \vee s', \\
&t \models_T s \text{ or } (t \rightarrow \& \ \forall t'.t \longrightarrow t' \Rightarrow t' \models_T \Box s) \Rightarrow t \models_T \Box s, \\
&t \models_T s \text{ or } \exists t'.t \longrightarrow t' \ \& \ t' \models_T \Diamond s \Rightarrow t \models_T \Diamond s.
\end{aligned}$$

Again, equivalently, one has $t \models_T \Box s$ iff all maximal branches from t meet a node satisfying s and $t \models_T \Diamond s$ iff there is a branch from t with a node satisfying s.

Again assume we are only interested in that information which holds inevitably, including statements like $\Box((\Diamond a) \vee b)$.

4.3 Definition. Let $(T, \longrightarrow, val)$ be a nondeterministic D–computation with root node t. Write $\models_T s$ for $t \models_T s$. Define

$$V_2(T) = \{ \Box s \in L_2 \mid \models_T \Box s \}.$$

Let T and T' be nondeterministic D–computations. Define

$$T \preceq_2 T' \Leftrightarrow V_2(T) \subseteq V_2(T').$$

Define

$$\simeq_2 = \preceq_2 \cap \preceq_2{}^{-1}.$$

4.4 Theorem. *Let $\mathcal{T}$ be the class of nondeterministic D–computations. The Plotkin powerdomain $P_2[D]$ is isomorphic to the quotient $(\mathcal{T}/\simeq_2, \preceq_2 / \simeq_2)$ and to the order $(\{ V_2(T) \mid T \in \mathcal{T} \}, \subseteq)$.*

Proof. (sketched) Again write $s \equiv s'$ iff $\forall T. \models_T s \Leftrightarrow \models_T s'$. We have the following equivalences: $\Diamond(\Diamond s) \equiv \Diamond(\Box s) \equiv \Box(\Diamond s) \equiv \Diamond s$; $\Box(\Box s) \equiv \Box s$; $\Diamond(s \vee s') \equiv (\Diamond s) \vee (\Diamond s')$; $\Box(s \vee (\Box s')) \equiv \Box(s \vee s')$; $\Box(s \vee (\Diamond s')) \equiv (\Box s) \vee (\Diamond s')$. Using these facts, a simple induction shows each sentence s of L_2 is equivalent to one s^* in normal form as shown:

$$\begin{aligned}
a_0 \vee \cdots \vee a_{n-1} \vee (\Diamond b_0) \vee \cdots \vee (\Diamond b_{m-1}) \vee \Box(c_0{}^0 \vee \cdots \vee c^0{}_{p_0 - 1}) \vee \cdots \\
\vee \Box(c_0{}^{q-1} \vee \cdots \vee c^{q-1}{}_{p_{q-1}-1}).
\end{aligned}$$

We use the convention that null sequences of statements represent $\perp \in D^0$. Observe in particular statements $\Box s$ have a normal form

$$\Box s \equiv \Box(a_0 \vee \cdots \vee a_{n-1}) \vee \Diamond b_0 \vee \cdots \vee \Diamond b_{m-1}.$$

Observe too that if $s \vee s' \in V_2(T)$ then $s \in V_2(T)$ or $s' \in V_2(T)$. From these two observations we obtain

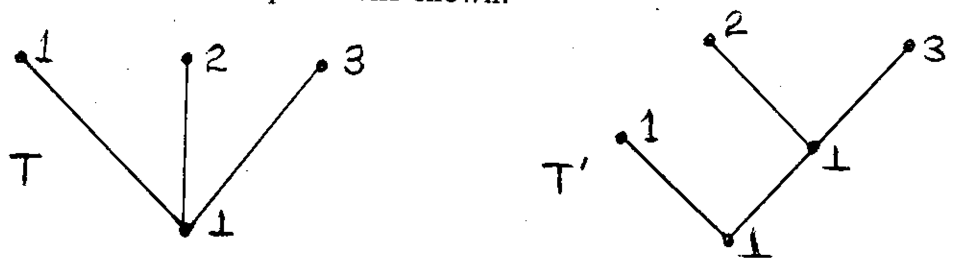
$$T \preceq_2 T' \Leftrightarrow V_0(T) \subseteq V_0(T') \;\&\; V_1(T) \subseteq V_1(T'').$$

The proof of "$\Rightarrow$" is clear. The proof of "$\Leftarrow$" follows from the two observations. Consequently, by theorems 2.5 and 3.4, we have $(\mathcal{T} / \simeq_2, \preceq_2 / \simeq_2) \cong P_2[D]$. ∎

Remark. It is clear from the proof that the result also holds if we restrict the language to sentences of the form $\Diamond a$ and $\Box(a_0 \vee \cdots \vee a_n)$ where $a, a_0, \ldots, a_n \in D^0$.

The next example shows the above results do not hold if one expands the language L_2 to include conjunction, with the obvious definition of satisfaction. A domain based on this expanded set of assertions must have a more complicated domain construction than that of a simple powerdomain, quite probably as a combination of powerdomain and product constructions.

4.5 Example. Assume N is the flat domain of integers with elements $\{\perp\} \cup \omega$ ordered by $x \sqsubseteq x$ and $\perp \sqsubseteq n$ for $n \in \omega$. Extend the language L_2 to include statements of the form $s \wedge s'$ and the satisfaction relation so $t \models_T s \wedge s'$ iff $t \models_T s$ and $t \models_T s'$. Let T and T' be the nondeterministic N–computations shown:



Let s be the statement $\Diamond(\Box(2 \vee 3) \wedge \Diamond 2 \wedge \Diamond 3)$. Then clearly $V_2(T) = V_2(T)$ but $\not\models_T s$ while $\models_{T'} s$; the introduction of $\wedge$ enables statements to express in more detail than L_2 how computations branch.

5. Generalisations.

It is not necessary to model the nondeterministic computations as trees. The above results would also hold if instead of trees we used transition systems. One could also allow states to be only partially defined to cope with divergence—see [HN]—provided one modified the definition of satisfaction for statements of the form $\Box s$. Such ideas are used in [HN] and [HM] to induce natural equivalences on communicating processes. It might be interesting to study the relation between domains of computations, like trees, and domains of sets of statements induced by satisfaction relations.

It would be interesting to relax the finitely-branching condition to allow ω-branching trees and in addition to the binary disjunction $\vee$ allow an infinitary disjunction $\bigvee$ over countable sets of assertions. It should be the case that one obtains versions of the Hoare and Plotkin powerdomains generalised to countable nondeterminism. The Plotkin powerdomain has been generalised to countable nondeterminism in [P2] using rather nonconstructive means, so a characterisaion based on modal assertions would provide a useful intuitive construction. Of course, if one follows the approach above and takes atomic statements to be the isolated or finite elements of a domain this construction would be of limited use because the resulting powerdomain would most likely not be algebraic, and so not amenable to the same construction itself. However, following this line may suggest a notion generalising that of isolated or finite element. Note that by the example of [AP] the obvious generalisation of section 2 to countable disjunction $\bigvee$ will not in general yield a c.p.o., so it can not directly yield a generalisation of the Smyth powerdomain to countable nondeterminism.

There are strong links with the idea of information systems a suggestive way of representing consistently complete algebraic c.p.o.'s presented by Dana Scott in [S]; elements of a domain are represented by consistent, deductively-closed sets of statements. However note until that framework is generalised to represent at least the SFP objects it will not support the Plotkin powerdomain construction. Still, it is an attractive idea, that the Scott-information denoting a computation can be regarded as the set of statements it satisfies. It invites us to look at domain constructions in a new way, as accompanying extensions to the languages with which we describe computations.

Acknowledgements.

This work was supported in part by Carnegie-Mellon University and in part by a postdoctoral fellowship from the Royal Society of Great Britain, to work at the Computer Science Department, Aarhus University, Denmark.

References

[AP] Apt, K.R. and Plotkin, G.D. A Cook's tour of Countable Nondeterminism. Lecture Notes in Comp. Sc. Vol.115, ICALP 1981.

[D] Dummett, M., Elements of Intuitionism. Oxford University Press (1977).

[Gue] Guessarian, I., Algebraic Semantics. Springer–Verlag LNCS Vol.99 (1981).

[HM] Hennessy, M.C.B. and Milner, R., On observing nondeterminism and concurrency, Springer LNCS Vol. 85. (1979).

[HN] Hennessy, M.C.B., and de Nicola, R., Testing Equivalences for Processes, Internal Report, University of Edinburgh, (July 1982).

[HP] Hennessy, M.C.B. and Plotkin, G., Full Abstraction for a Simple Parallel Programming Languages, LNCS Vol 74 (1979).

[Ho] Hoare, C. A. R., A Model for Communicating Sequential Processes, Technical Report PRG–22, Programming Research Group, Oxford University Computing Lab. (1981).

[LO] Lamport, L., and Owicki, S., Proving Liveness Properties of Concurrent Programs, Technical Report, SRI International (1980).

[P1] Plotkin, G., A powerdomain construction, SIAM J. on computing, 5, pp.452–486, 1976.

[P2] Plotkin, G., A Powerdomain for countable non–determinism, Springer–Verlag Lecture Notes in Comp. Sc. 140 (1982).

[Smy] Smyth, M., Powerdomains. JCSS Vol.16 No.1 (1978).

[S] Scott, D., Domains for Denotational Semantics, Springer–Verlag Lecture Notes in Comp. Sc. 140 (1982).