An Experiment with Domain Construction for Denotational Semantics

by

Wihlinne's Debrief
AN EXPERIMENT WITH DOMAIN CONSTRUCTION
FOR DENOTATIONAL SEMANTICS
(extended abstract)

Włodzimierz Drabent

Department of Computer and Information Science
Linköping University, 581 83 Linköping, Sweden

on leave from

Institute of Computer Science
Polish Academy of Sciences
P.O.Box 22, 00-901 Warszawa PKiN, Poland

ABSTRACT. A construction of domains for denotational semantics is proposed. In the author's opinion it is simpler, more natural and more comprehensible than the traditional $P \omega$ construction. The basic idea of the approach is to determine a domain element through its finite, "representable-in-computer", approximations. Domains are built from so called kits which are quasi-ordered sets of such approximations. Informally, the quasi-order reflects the "amount of information" carried by single kit elements. The domain determined by a kit is the set of ideals which are certain subsets of that kit. We define kit operations which are counterparts of traditional domain operations (sum, product, lift, function space). We present some methods of generating functional domain elements by their small but representative subsets. The kit operations may be used to construct kit equations which are counterparts of domain equations. We discuss kit equation solving, including those yielding counterparts of reflexive domains.

The paper was motivated by a project concerning the construction of operational definitions which are very close to denotational ones. Kit elements are approximations which can be produced by such an operational definition "one at a time".
0. INTRODUCTION

Mathematical foundations are the most difficult part of denotational semantics. Traditional approaches (see [S] and references there) are not easily comprehensible. Some textbooks (e.g., [C]) skip the foundations at all. It seems that many practitioners do not understand them.

So there is need to seek other approaches to the theory of denotational semantics which may eventually help in presenting the theory in a more comprehensible and more intuitive way. D. Scott has himself presented four constructions (see [S], [Sc] and references there). Blikle and Tarlecki [BT] discussed programming language constructs which may be described without using reflexive domains and present a variant of denotational semantics using sets instead of domains.

This paper also deals with this problem. The approach presented here is based on the fact that domains are required to have countable bases. The main idea is to define a domain by supplying a set of "building bricks" from which domain elements are to be built. Such a set, augmented with a quasiorder, is called a "kit". Its elements may be understood as finite approximations of domain elements. The intuitive meaning of the quasiorder is that \( a \sqsubseteq b \) when information carried by \( b \) implies information carried by \( a \). Domain elements are certain subsets of a kit (so called ideals). They can be generated by increasing chains of kit elements; an element of domain basis may be generated by a single kit element. As a kit fully determines the corresponding domain, most of reasonings concerning domains may be carried out in terms of kits. This concerns, among others, domain operations and domain equation solving.

A close connection between the approach presented here and information systems of Scott [Sc][LW] should be pointed out. The basic idea of information systems is the same as that of our approach: domain elements are certain subsets of a given set. For information systems it is a set of propositions [Sc] or tokens [LW]. Kit quasiorder corresponds to entailment relation of information systems. Kit elements play the role of both tokens and consistent sets. Both approaches define the same class of domains, the class of consistently complete algebraic c.p.o.s (see example 4 at the end of section 1).

The basic difference is that instead of the three stage construction given by information systems (tokens - consistent sets - domain elements) we present a two stage one (kit elements - domain elements). Therefore definitions of kit operations and relations are usually simpler than the corresponding ones for information systems. There is also a clear
difference between intuitions connected with the two approaches. In information systems a domain element is a set of "propositions" about itself, whereas in our approach it is a set of its "finite approximations". Another difference is that to define function space we use finite functions (between kits) instead of finite relations (between sets of consistent sets). Another contribution of this paper is the discussion on representative subsets of functional domain elements; a discussion which seems to be easily transferable to the information systems framework.

Our approach is connected with a semantics description method which is both operational and denotational [D]. The method assigns meanings in an operational (and nondeterministic) way, they are finite approximations of a (possibly infinite) denotational meaning. Arbitrarily good approximations of the denotational meaning may be generated. The approximations correspond to kit elements, the set of all approximations given as meaning for a language construct corresponds to domain element.

The paper is organized as follows. Kits and domains generated by kits are defined in section 1. In sections 2 and 3 we introduce kit operations corresponding to domain constructors in denotational semantics: disjoint sum, product, lift, function space. As functional domain elements are rather "big" sets of kit elements, we show in section 4 three methods of generating them by their "small" but representative subsets. These subsets may be usually specified in a simple and natural way as it is shown by the examples at the end of this section. In section 5 we discuss kit equation solving. It is worth mentioning that we solve "exact" equations like \(X = F(X)\) instead of isomorphic ones (like \(X \cong F(X)\)). As an example, an equation defining a reflexive domain which may occur in a denotational description of a programming language is presented. Relationship between individual elements of the domain and intuitions connected with its use in programming language descriptions are exemplified. The result is, in the author's opinion, simpler and more intuitive than the traditional \(P\omega\) construction [S].

1. DOMAIN CONSTRUCTION

Preliminary notions. By a c.p.o. we mean a chain complete partial order. A function is said to be continuous if it preserves l.u.b.s of increasing chains. A reflexive and transitive relation is called a quasi-ordering (q-ordering). For example every partial ordering and every equivalence relation is a q-ordering.
Let \((B, \sqsubseteq)\) be a q-ordered set.

**Definition.** \(X \subseteq B\) is an ideal in \((B, \sqsubseteq)\) provided that

(i) \(b \in X \& b' \sqsubseteq b \implies b' \in X,\)

(ii) \(b_1, b_2 \in X \implies (\exists b \in X) b_1, b_2 \sqsubseteq b.\)

**Remark.** Wherever it does not lead to misunderstandings we will denote \((B, \sqsubseteq)\) by \(B\), etc. and use the same symbol for similar relations in different sets.

**Examples of ideals:**

\[
\emptyset, \\
\{b\} \quad \text{which is, by definition,} \quad \{b' \in B \mid b' \sqsubseteq b\}, \\
\bigcup_i b_i \quad \text{where} \quad b_1 \sqsubseteq b_2 \sqsubseteq \ldots.
\]

If \(B\) is countable then these are the only ideals in \(B\).

**Definition.** \(\mathcal{D}(B, \sqsubseteq) = \{X \subseteq B \mid X\) is an ideal in \((B, \sqsubseteq)\}\).

According to our convention we will often use the abbreviation \(\mathcal{D}(B)\).

**Proposition.** \((\mathcal{D}(B), \sqsubseteq)\) is a c.p.o. with \(\emptyset\) as the least element.

**Definition.** \(b_1, \ldots, b_n \in B\) are consistent iff there exists \(b \in B\) such that \(b_1, \ldots, b_n \sqsubseteq b\).

**Definition (of join for a q-order).** Let \(b_1, b_2 \in B.\)

\[
b_1 \sqcup b_2 = \{b \in B \mid b_1, b_2 \sqsubseteq b \quad \& \quad (\forall c \in B) b_1, b_2 \sqsubseteq c \implies b \sqsubseteq c\}.
\]

Of course, if \(c, c' \in b_1 \sqcup b_2\) then \(c \sqsubseteq c' \sqsubseteq c\).

**Definition.** \((B, \sqsubseteq)\) is a kit iff

\[
B \text{ is countable,} \\
\sqsubseteq \text{ is a q-order,} \\
\text{the join of every two consistent elements of } B \text{ is nonempty.}
\]

**DEFINITION.** A domain is every \(\mathcal{D}(B)\), where \(B\) is a kit.

Domains have so called finite elements which are \(\emptyset\) and all \(!b\) where \(b \in B\). The remaining elements are infinite and each is a union \(\bigcup_i b_i\) where \((b_i)\) is an increasing chain in \(B\). (Note: finite elements of a domain need not to be finite sets).
Examples.

1. The domain built from the kit \((B, =)\) is a flat one: \(\mathcal{D}(B) = \{\emptyset\} \cup \{\{b\} \mid b \in B\}\).

2. Let \(B_1, B_2\) be sets, \(B = \{f : B_1 \to B_2 \mid f\) finite and nonempty \}. The set-theoretical inclusion \(\subseteq\) is a q-ordering in \(B\) with the join \(f \sqcup f' = \{f \sqcup f'\}\) for every consistent \(f, f' \in B\). Ideals in \((B, \subseteq)\) are \(\{f \in B \mid f \subseteq g\}\) for every (partial) function \(g : B_1 \to B_2\). For \(g\) being total function we obtain a maximal element of \(\mathcal{D}(B)\).

3. Let \((D, \sqsubseteq)\) be a Scott domain with the basis \((E, \sqsubseteq)\) (according to definitions of Stoy [S]). \((E, \sqsubseteq)\) is a kit with the join \(e \sqcup e' = \{\text{l.u.b.}\{e, e'\}\}\) for every \(e, e' \in E\). Every \(h(x) = \{e \in E \mid e \sqsubseteq x\}\) (for \(x \in D\)) is an ideal in \(E\), \(h\) is a monotonic and one-to-one function from \(D\) to \(\mathcal{D}(E)\). It is also a function onto \(\mathcal{D}(E)\), under the assumption that if \(e \sqsubseteq \text{l.u.b.} Y\) then \((\exists e' \in Y)\) \(e \sqsubseteq e'\) (for every \(e \in E\), \(Y \in \mathcal{D}(E)\)). (This requirement is equivalent to finiteness of basis elements. It seems that it is fulfilled by every domain needed in practice.) The \(h^{-1}\) \((h^{-1}(Y) = \text{l.u.b.} Y)\) is also monotonic, thus \(h\) is a lattice isomorphism and \((\mathcal{D}(E), \subseteq), (D, \sqsubseteq)\) are isomorphic.

4. Let \(A = (A, \Delta, \text{Con}, \vdash)\) be an information system [Sc] and let \(A\) be countable. Let \(\vdash\) be the entailment relation between \(\text{Con}\) and \(\text{Con}\) (see [Sc], definition 2.2). Then \((\text{Con}, \vdash)\) is a kit and it generates a domain isomorphic to that generated by \(A\).

Let \((B, \sqsubseteq)\) be a kit. To transform it into an information system, let \(\{b_1, \ldots, b_n\} \in \text{Con}\) iff \(b_1, \ldots, b_n\) are consistent in \((B, \sqsubseteq)\) and let \(\{b_1, \ldots, b_n\} \vdash b\) iff \(b \sqsubseteq b_1 \sqcup \ldots \sqcup b_n\) (see the discussion on multiple joins at the beginning of section 3). Then \((B, \text{Con}, \vdash)\) is an information system [LW] and it generates the same domain as does the kit \((B, \sqsubseteq)\).

2. SIMPLE KIT OPERATIONS

Domain operations are defined by means of corresponding kit operations. First we introduce product \(\otimes\), disjoint union \(\oplus\) and lift \(\circ\). In the next section we define \(\otimes\) kit construction operation for the functional domain.

Definition. Let \(B, B_1, B_2\) be q-ordered sets.

1. \(B_1 \otimes B_2 = B_1 \times B_2\) with the q-ordering

\[(b_1, b_2) \sqsubseteq (b'_1, b'_2)\] if and only if \(b_i \sqsubseteq b'_i\) in \(B_i\) for \(i = 1, 2\).

2. \(B_1 \oplus B_2 = \{1\} \times B_1 \cup \{2\} \times B_2\) with the q-ordering

\[(i, b) \sqsubseteq (j, b')\] if and only if \(i = j\) \& \(b \sqsubseteq b'\) in \(B_i\).
3. $\ominus B = B \cup \{ b_0 \}$, where $b_0 \notin B$, with the q-ordering

$$b \sqsubseteq b' \quad \text{iff} \quad b = b_0 \; \text{or} \; b \sqsubseteq b' \; \text{in} \; B.$$ 

**Proposition.**

If $B_1, B_2, B$ are kits then $B_1 \oplus B_2, B_1 \ominus B_2, \ominus B$ are kits.

$$\mathcal{D}(B_1 \ominus B_2) = \{ X_1 \times X_2 \mid X_1 \in \mathcal{D}(B_1) \; \& \; X_2 \in \mathcal{D}(B_2) \}$$

$$\mathcal{D}(B_1 \oplus B_2) = \{ \{1\} \times X \mid X \in \mathcal{D}(B_1) \} \cup \{ \{2\} \times X \mid X \in \mathcal{D}(B_2) \}$$

$$\mathcal{D}(\ominus B) = \{ \emptyset \} \cup \{ \{b_0\} \cup X \mid X \in \mathcal{D}(B) \}$$

The union and product defined above are "coalescent". When this is inconvenient we may use $(\ominus B_1) \ominus (\ominus B_2)$ and $(\ominus B_1) \oplus (\ominus B_2)$.

It is easy to show that the operations $\ominus, \oplus, \ominus$ are continuous in an appropriate lattice (with q-ordered sets as elements) ordered by pointwise inclusion as follows:

$$(B_1, \sqsubseteq_1) \subseteq (B_2, \sqsubseteq_2) \quad \text{iff} \quad B_1 \subseteq B_2 \; \& \; \sqsubseteq_1 \subseteq \sqsubseteq_2.$$ 

**Examples.** The continuity of these operations allows solving kit equations. This is illustrated by some examples. Within the examples we use the simplified version of $\ominus$ for disjoint sets. If $B_1 \cap B_2 = \emptyset$, we define $B_1 \ominus B_2$ as $B_1 \cup B_2$ with an appropriate q-ordering. We treat the cartesian product as associative and write $(a, b, c)$ instead of $\langle (a, b), c \rangle$ etc.

It may be checked that equation solutions in the examples are kits. This fact follows also from propositions in the paragraph 5 where kit equation solving is discussed.

**Ex. 1.** $X = B \oplus (B \ominus X)$

We are looking for the least fixed point (l.f.p.) of the function $F$, $F(X) = B \oplus (B \ominus X)$.

$$F^n(\emptyset) = \bigcup_{i=1}^n B^i$$

with the q-ordering

$$(b_1, \ldots, b_k) \sqsubseteq (b'_1, \ldots, b'_l) \quad \text{iff} \quad k = l \; \& \; b_i \sqsubseteq b'_i \; \text{for} \; i = 1, \ldots, k.$$

Thus the l.f.p. is $X = \bigcup_i B^i$ with this q-ordering. $\mathcal{D}(X)$ is a domain of finite sequences over $\mathcal{D}(B)$ because if $V \in \mathcal{D}(X)$ then $(\exists k \exists V_1, \ldots, V_k \in \mathcal{D}(B)) \; V = V_1 \times \cdots \times V_k$.

**Ex. 2.** $X = B \ominus \ominus X$.

We are looking for l.f.p. of $F$ where $F(X) = B \ominus \ominus X$.

$$F^n(\emptyset) = \bigcup_{i=1}^n B^i \times \{ 0 \}$$

(where $0 \notin B$) with the q-ordering

$$(b_1, \ldots, b_k, 0) \sqsubseteq (b'_1, \ldots, b'_l, 0) \quad \text{iff} \quad k \leq l \; \& \; b_i \sqsubseteq b'_i \; \text{for} \; i = 1, \ldots, k.$$
Thus the solution is $X = \bigcup_i B^i \times \{0\}$ with this q-ordering. \(\mathcal{D}(X)\) is a domain of finite and infinite sequences over \(\mathcal{D}(B)\): if \(V \in \mathcal{D}(X)\) then either ($\exists k \exists V_1, \ldots, V_k \in \mathcal{D}(B)$) $V = V_1 \times \cdots \times V_k \times \{0\}$ or ($\exists V_1, V_2, \ldots \in \mathcal{D}(B)$) $V = \bigcup_k V_1 \times \cdots \times V_k \times \{0\}$.

3. FUNCTIONAL KITS

To define $\Theta$ we introduce some additional notions. We will use signs $\subseteq$ and $\sqcup$ in new contexts. We will write "$b_1 \sqcup b_2 \sqsubseteq c"$ instead of "($\exists d \in b_1 \sqcup b_2$) $d \sqsubseteq c", \text{etc. We will also write "}(b_1 \sqcup b_2) \sqcup b_3" \text{instead of } "d \sqcup b_3 \text{where } d \in b_1 \sqcup b_2", \text{etc. We define}

$$b_1 \sqcup \cdots \sqcup b_n = \{ b \in B \mid b_1, \ldots, b_n \sqsubseteq b \& (\forall c)(b_1, \ldots, b_n \sqsubseteq c \Rightarrow b \sqsubseteq c) \}. $$

Of course, $b_1 \sqcup \cdots \sqcup b_n = (\ldots(b_1 \sqcup b_2) \sqcup \ldots) \sqcup b_n$. For $b_1, \ldots, b_n$ being elements of a kit, $b_1, \ldots, b_n$ are consistent iff $b_1 \sqcup \cdots \sqcup b_n$ is nonempty.

The argument set of a function \(f\) will be denoted by \(\text{arg} f\). Finite functions will be called mappings. The set of all mappings from \(B\) to \(C\) will be denoted \(B \overset{m}{\rightarrow} C\). The notation we use for mappings is that of the Vienna Development Method.

**Definition.** A mapping $m: B \overset{m}{\rightarrow} C$ is correct provided that it gives consistent results for consistent arguments:

$$ (\forall b_1, \ldots, b_n \in \text{arg} m)(b_1, \ldots, b_n \text{ are consistent} \Rightarrow m(b_1), \ldots, m(b_n) \text{ are consistent}). $$

**Definition** (of the q-ordering for correct mappings). Let \(B, C\) be kits, $m, m': B\overset{m}{\rightarrow} C$ be correct mappings. $m \sqsubseteq m'$ iff

$$ (\forall b \in \text{arg} m)(\exists n \exists b_1, \ldots, b_n \in \text{arg} m')(b \sqsubseteq b_1, \ldots, b_n \& m(b) \sqsubseteq m'(b_1) \sqcup \ldots \sqcup m'(b_n)) $$

(intuitively: \(m'\) gives greater results for lesser arguments).

**Proposition.** The above defined $\sqsubseteq$ is a q-ordering and the following holds for every correct $m, m': B \overset{m}{\rightarrow} C$.

1. \(m \sqsubseteq m' \Rightarrow m \sqsubseteq m'\)
2. \((\forall b \in \text{arg} m)(b \in \text{arg} m' \& m(b) \sqsubseteq m'(b)) \Rightarrow m \sqsubseteq m'\)
3. \((\forall b \in \text{arg} m \exists b' \in \text{arg} m')(b \sqsupseteq b' \& m(b) \sqsubseteq m'(b')) \Rightarrow m \sqsubseteq m'\)
4. \(b \sqsupseteq b' \& c \sqsubseteq c' \Rightarrow [b \rightarrow c] \sqsubseteq [b' \rightarrow c'], \text{ where } b, b' \in B, c, c' \in C\)
5. \(b \sqsubseteq b_1 \sqcup b_2 \& c \in c_1 \sqcup c_2 \& m_1 = [b_1 \rightarrow c_1] \sqcup [b_2 \rightarrow c_2] \& m_2 = m_1 \sqcup [b \rightarrow c] \Rightarrow [b \rightarrow c] \sqsubseteq m_1 \sqsubseteq m_2 \sqsubseteq m_1, \text{ where } b, b_1, b_2 \text{ are distinct elements of } B, c, c_1, c_2 \in C.\)
Definition (of functional kit). Let $B, C$ be kits.

$$B \otimes C = \{ m : B \Rightarrow C \mid m \text{ is nonempty and correct} \}$$

with the q-ordering as above.

Proposition. $B \otimes C$ is a kit.

To prove this, the nonemptiness of the join of every two consistent $m_1, m_2 \in B \otimes C$ must be shown. Let $m_1, m_2 \subseteq m$, $\text{argm}_i = X_i$ for $i = 1, 2$. Let us define

$$(m_1 \otimes m_2)(b) = \begin{cases} m_1(b) & \text{for } b \in X_1 \setminus X_2 \\ m_2(b) & \text{for } b \in X_2 \setminus X_1 \\ c & \text{for } b \in X_1 \cap X_2, \text{ where } c \in m_1(b) \sqcup m_2(b). \end{cases}$$

Now we can prove that such $c$ exist (for every $b \in X_1 \cap X_2$), that $m_1 \otimes m_2$ is a correct mapping and that $m_1 \otimes m_2 \subseteq m$. This implies $m_1 \otimes m_2 \in m_1 \sqcup m_2$. We omit details of the proof.

Now we establish the connection between continuous functions from $\mathcal{D}(B)$ to $\mathcal{D}(C)$ and elements of $\mathcal{D}(B \otimes C)$. Let $X \in \mathcal{D}(B)$, $U \in \mathcal{D}(B \otimes C)$.

Definition (of functional application).

$$UX = \{ c \in C \mid (\exists b \in X \exists m \in U) \ c \subseteq m(b) \}.$$  

Proposition (which may be treated as an alternative definition of the functional application).

$$UX = \{ c \in C \mid (\exists b \in X)[b \rightarrow c] \in U \}.$$  

Proposition. Let $m \in B \otimes C$.

$$(!m)X = \{ c \in C \mid (\exists b_1, \ldots, b_n \in X \cap \text{argm}) \ c \subseteq m(b_1) \sqcup \ldots \sqcup m(b_n) \}.$$  

Proposition. $UX \in \mathcal{D}(C)$ and the function $\lambda X. UX$ from $\mathcal{D}(B)$ to $\mathcal{D}(C)$ defined by $U$ is strict and continuous.

Proposition. Let $U, V \in \mathcal{D}(B \otimes C)$.

$$U \subseteq V \text{ iff } UX \subseteq VX \text{ for every } X \in \mathcal{D}(B)$$

(the ordering in $\mathcal{D}(B \otimes C)$ is the same as the usual ordering in a functional domain).
Proposition. Let \( f : \mathcal{D}(B) \rightarrow \mathcal{D}(C) \) be a strict and continuous function. Let
\[
I(f) = \{ m \in B \otimes C \mid (\forall b \in \text{arg} m) \, m(b) \in f(b) \}.
\]

Then
\[
I(f) \in \mathcal{D}(B \otimes C) \quad \text{and} \quad (I(f))x = f(x).
\]

Proposition. Let \( U, V \in \mathcal{D}(B \otimes C) \), \( U \neq V \). Then the functions defined by \( U \) and \( V \) are different (there exists \( b \in B \) such that \( U(b) \neq V(b) \)).

CONCLUSION. To every strict and continuous function from \( \mathcal{D}(B) \) to \( \mathcal{D}(C) \) there corresponds exactly one element of \( \mathcal{D}(B \otimes C) \) (and vice versa).

The exclusion of nonstrict functions is insignificant because every function \( f : \mathcal{D}(B) \rightarrow \mathcal{D}(C) \) has a corresponding strict function \( f' : \mathcal{D}(\otimes B) \rightarrow \mathcal{D}(C) \), \( f'(X \cup \{b_0\}) = f(X) \) for every \( X \in \mathcal{D}(B) \) (\( b_0 \notin B \)).

Comment. The need to obtain this one-to-one correspondence led us to q-orderings (instead of orderings) and to the presented form of \( \subseteq \) for mappings. If \( \subseteq \) had not had the properties 3 and 4 there might have existed many ideals defining the same function.

4. GENERATION OF FUNCTIONAL DOMAIN ELEMENTS

Elements of a functional domain are often "big" sets. In this section we show how these sets and their associated functions can be uniquely determined by their "small" subsets. We define three generating operations. For their arguments we present the conditions which guarantee obtaining functional domain elements as results. We also show how to define the functional application in terms of these "small" subsets. The section is completed by some examples.

Let \( B, C \) be kits, \( W \subseteq \{ [b \rightarrow c] \mid b \in B \& c \in C \} \).

Definition.
\[
G_0(W) = \{ [b_1 \rightarrow c_1, \ldots, b_n \rightarrow c_n] \mid [b_i \rightarrow c_i] \in W \text{ for } i = 1, \ldots, n \}.
\]

Proposition. \( G_0(W) \in \mathcal{D}(B \otimes C) \) iff
\[
\begin{align*}
1. & \quad [b' \rightarrow c'] \subseteq [b \rightarrow c] \in W \Rightarrow [b' \rightarrow c'] \in W \\
2. & \quad [b \rightarrow c_1], [b \rightarrow c_2] \in W \Rightarrow c_1, c_2 \text{ are consistent } \& [b \rightarrow c] \in W \text{ for } a \in c_1 \cup c_2.
\end{align*}
\]
Definition. \[ G_1(W) = G_0(\{ [b \mapsto c'] \mid (\exists [b \mapsto c] \in W) [b \mapsto c'] \subseteq [b \mapsto c] \}). \]

Proposition. \( G_1(W) \in D(B \odot C) \) iff \([b_1 \mapsto c_1], [b_2 \mapsto c_2] \in W \) and \( b_1, b_2 \) are consistent \( \implies \) \((\exists c \supseteq b_1 \cup b_2, \exists c \supseteq c_1 \cup c_2) [b \mapsto c] \in W. \)

Proposition. Let \( X \in D(B) \), \( G_1(W) \in D(B \odot C) \). Then \((G_1(W))X = \{ c' \mid (\exists [b \mapsto c] \in W) b \in X & c' \subseteq c \}. \)

Definition. \[ G_2(W) = G_0(\{ [b \mapsto c] \mid (\exists b \supseteq b_1, \exists c \supseteq c_1) [b \mapsto c] \in W \}). \]

Proposition. \( G_2(W) \in D(B \odot C) \) iff \[ 1. [b_1 \mapsto c_1] \in W \text{ for } i = 1, \ldots, n \text{ and } b_1, \ldots, b_n \text{ are consistent} \implies c_1, \ldots, c_n \text{ are consistent}, \]

Proposition. Let \( X \in D(B) \), \( G_2(W) \in D(B \odot C) \). Then \((G_2(W))X = \{ c \mid (\exists [b \mapsto c] \in W) b \in X \text{ and } c \subseteq c_1 \cup \ldots \cup c_n \}. \)

Examples.

1. Let \( A, B \) be kits, \( U = \{ ([1, a] \mapsto a') \mid a, a' \in A \text{ and } a' \subseteq a \}. \) Then \( G_0(U) \in D(A \oplus B \odot A) \), \((G_0(U))([1] \times X) = X \) for \( X \in D(A) \) and \((G_0(U))([2] \times Y) = \emptyset \) for \( Y \in D(B) \). So \( G_0(U) \) is a projection operation from \( D(A \oplus B) \) into \( D(A) \).

2. Let \( B, C \) be kits and \( \text{apply} = G_1(\{ ([b \mapsto c], b \mapsto c] \mid b \in B, c \in C \}). \) Then \( \text{apply} \in D((B \odot C) \odot B \odot C) \) and \( \text{apply}(V \times X) = \{ c' \mid (\exists [b \mapsto c] \in V) b \in X \text{ and } c' \subseteq c \} = VX. \)

3. Let \( A, B, C \) be kits and \( \text{curry} = G_2(\{ [[[a, b] \mapsto c]], [a \mapsto [b \mapsto c]]] \mid a \in A, b \in B, c \in C \}). \) It may be checked that \( \text{curry} \in D((A \oplus B \odot C) \odot A \odot B \odot C) \) and that \( \text{curry}VTU = V(T \times U) \) for \( V \in D(A \odot B \odot C), T \in D(A), U \in D(B) \).
Let $A$ be kit, $B = \Theta A$, $0 \in B \setminus A$,
$$F = \{[m \rightarrow x] \mid (\exists x_1, \ldots x_n \in B) \ x_n = x \ & \ & m = [0 \rightarrow x_1, x_1 \rightarrow x_2, \ldots, x_{n-1} \rightarrow x_n]$$
& $m$ is correct \}.

Let $\text{fix} = G1(F)$ and $U \in D(B \circ B)$. It may be checked that $\text{fix} \in D((B \circ B) \circ B)$, $U(\text{fix}U) = \text{fix}U$ and if $X \in D(B)$, $X \neq \emptyset$ and $UX \subseteq X$ then $\text{fix}U \subseteq X$. So fix gives the least fixpoint for a given continuous function from $D(A)$ to $D(A)$ (because the function may be nonstrict, it is represented by the corresponding function from $D(B)$ to $D(B)$).

5. KIT EQUATIONS INVOLVING $\Theta$

The operation $\Theta$ is not monotonic with respect to its first argument in a set of kits ordered by pointwise inclusion (because nonconsistent elements of $B$ may be consistent in $B' \supseteq B$; in consequence a mapping may be correct when treated as from $B$ and not correct when treated as from $B'$). We introduce a new ordering $\leq$ for kits to make all the four kit operations continuous with respect to every argument.

Definition. Let $X = (X, \sqsubseteq), Y = (Y, \sqsubseteq)$ be kits.

$\mathbf{X} \leq \mathbf{Y}$ iff

1. $X \subseteq Y$ \& $\sqsubseteq \subseteq \sqsubseteq$,
2. $(\forall a, b \in X) \ a \sqcup b \sqsubseteq a \sqcup b \ \& \ (a \sqcup b \neq \emptyset \Rightarrow a \sqcup b \neq \emptyset)$

(where $\sqcup$ is the join in $X$ and $\sqcup$ is the join in $Y$).

Note. Condition 2 above implies that if $a_1, \ldots, a_n$ are consistent in $Y$ then they are consistent in $X$.

Lemma. Let $X_1 \leq X_2 \leq \ldots$, where $X_i = (X_i, \sqsubseteq_i)$ is a q-ordered set for $i = 1, 2, \ldots$, and let $\sqcup_i$ be the join in $X_i$ and $\sqcup$ be the join in $\bigcup_i X_i = (\bigcup_i X_i, \sqsubseteq_i)$. Then $a \sqcup b = \bigcup_{i \geq k} (a \sqcup_i b)$ for every $a, b \in \bigcup_i X_i$, where $k$ is such that $a, b \in X_k$. If $X_i$ is a kit for $i = 1, 2, \ldots$ then $\bigcup_i X_i$ is a kit.

Proposition. Let $A$ be a set of kits and let $X_1, X_2, \ldots \in A$, $\bigcup_i X_i \in A$. With the assumptions of the previous lemma

$$\bigvee_i X_i = \bigcup_i X_i$$

(where $\bigvee$ is l.u.b. in $(A, \leq)$).
We omit the proof. From the proposition it follows that a function continuous w.r.t.
\( \subseteq \) and monotone w.r.t. \( \preceq \) is also continuous w.r.t. \( \preceq \).

**Proposition.** The operations \( \Theta, \Theta, \Theta, \Theta \) are continuous in every \((A, \preceq)\) where \( A \), a
set of kits, is closed under these operations and under set theoretical unions of increasing
chains.

Existence of such \((A, \preceq)\) that \( A \) contains a given set of basic kits follows from axioms
of set theory (see the discussion of similar problem in [BT]). As one of the basic kits we may
assume \((0,0)\), thus obtaining \((A, \preceq)\) which is a c.p.o. of kits. The formula for equation
solving in such c.p.o. is the same as in the lattice of q-ordered sets. Namely, in both cases
the l.f.p. of a continuous function \( F \) (with arguments and results being kits) is \( \bigcup \{ F^i(0,0) \} \).

**Remark.** Of course, the previous examples of kit equation solving remain valid in a c.p.o.
of kits ordered by \( \preceq \). It is worth mentioning that we solve kit equations not only up to
isomorphism.

**Example.**

Consider the equation \( P = \Theta P \Theta T \), where \( T \) is a kit. As in the previous examples:

\[
F(0) = \{0\} \Theta T
\]

\[
F^2(0) = \Theta(\{0\) \Theta T) \Theta T
\]

\[
F^3(0) = \Theta(\Theta(\{0\) \Theta T) \Theta T) \Theta T
\]

\[
\ldots
\]

\[
F^n(0) = \Theta(\ldots \Theta(\ldots (\{0\) \Theta T) \Theta T) \Theta T)n \text{ times}
\]

with the usual ordering for correct mappings. The l.f.p. is \( P = \bigcup \{ F^n(0) \} \).

A comment: Assuming that \( T = S \Theta S \), where \((S,=)\) is the kit of states, \( D(P) \) may
be considered as a semantic domain for procedures with one procedural parameter. Let
\( U \in D(P) \) and let \([0\rightarrow[s\rightarrow s']] \in U \). This means that execution of the procedure \( U \) begun
at state \( \{s\} \) ends at \( \{s'\} \) no matter what the parameter is (the parameter procedure is not
called during the execution). An interpretation of \([p\rightarrow[s\rightarrow s']] \in U \) is that the procedure
\( U \) called at the state \( \{s\} \) with a parameter \( V \equiv p \) ends at state \( \{s'\} \).

Now we construct an interesting element of \( P \). Let \( t_1, t_2, \ldots \in T \), \( t_i \subseteq t_{i+1} \) and \( t_i \not\supseteq t_{i+1} \) for \( i = 1, 2, \ldots \), \( p_1 = 0 \), \( p_i = [p_{i-1} \rightarrow t_{i-1}] \) for \( i = 2, 3, \ldots \), \( m_i = [p_1 \rightarrow t_1, \ldots, p_i \rightarrow t_i] \) for \( i = 1, 2, \ldots \). Then, for every \( i = 1, 2, \ldots \), \( m_i \) is correct, \( m_i \subseteq m_{i+1} \) and \( m_i \not\supseteq m_{i+1} \).
(because \( t_{i+1} \not\subseteq t_1 \cup \ldots \cup t_i \)). Then \( \bigcup_i \cdot t_i \) is an element of \( D(P) \) and is not an element of any \( D(F^n(\emptyset)) \).

End of the example

ACKNOWLEDGEMENTS. I am indebted to Jacek Leszczyłowski for discussions during the preparation of this paper. I would like to thank Gordon Plotkin, Jan Małuszynski and Neil D. Jones for their critical comments. Barbara Kopeć and Kevin Ryan helped to improve the English of this paper. A major part of this work was done at the Institute of Computer Science of Polish Academy of Sciences.

REFERENCES.


[Sc] Scott, D.S., "Domains for denotational semantics", ICALP 82, Springer Lecture Notes in Computer Science 140, p.577-613

A reference added in proof:

ABSTRACT. A construction of domains for denotational semantics is proposed. In the author's opinion it is simpler, more natural and more comprehensible than the traditional P construction. The basic idea of the approach is to determine a domain element through its finite, "representable-in-computer", approximations. Domains are built from so-called kits which are quasi-ordered sets of such approximations. Informally, the quasi-order reflects the "amount of information" carried by single kit elements. The domain determined by a kit is the set of ideals which are certain subsets of that kit. We define kit operations which are counterparts of traditional domain operations (sum, product, lift, function space). We present some methods of generating functional domain elements by their small but representative subsets. The kit operations may be used to construct kit equations which are counterparts of domain equations. We discuss kit equation solving, including those yielding counterparts of reflexive domains.