G. HUET

DATA TYPES AS LATTICES

OR

THE STUDY OF THE
LOGICAL TYPES
OF
STRUCTURED DATA

Lectures by

DANA SCOTT

AMSTERDAM
June 1972

CHAPTER I. Basics

- MOTTO: Extend BNF to Semantics.
- WARNING: Distinguish Mathematical Semantics from Implementation

BRIEF EXPLICATION: A <u>data type</u> is a domain (usually: D) of data structures grouped together in order to be able to discuss a body of general <u>operations</u> defined on all objects of that type. If φ is such an operation, we might write:

$$\varphi : D' \times D'' \times D''' \rightarrow D$$

to indicate that φ has three arguments (of the indicated types) and takes values in D. This statement shows the <u>logical type</u> of the operation φ. Operations (of a given type) can be grouped to form a new data type also.

REFERENCES & BACKGROUND. The motivation for using lattices has been detailed in a series of papers referenced in: "Mathematical concepts in programming language semantics", AFIPS Conf. Proc. vol 40 (Spring Joint 1972). The main idea is to have a flexible, general, and natural theory of <u>partial objects</u> <u>approximations</u>, & <u>recursive definitions</u>. This seems to be possible by using a few concepts from lattice theory such as <u>continuous functions</u> and <u>fixed-point operators</u>. There are also close connections with topology as shown in the paper "Continuous Lattices." For such projects as constructing models of the λ-calculus (a very simple type of programming language) there does not seem to be any other approach.

DEFINITION. A *partially ordered set* is a set D of elements together with a relation $\sqsubseteq$ such that for all $x, y, z \in D$ we have:

(1) $x \sqsubseteq x$

(2) $x \sqsubseteq y \sqsubseteq z$ implies $x \sqsubseteq z$

(3) $x \sqsubseteq y \sqsubseteq x$ implies $x = y$

DEFINITION. A *lattice* is a partially ordered set $D, \sqsubseteq$ for which there exist elements $\perp, \top$ and binary operations $\sqcup$ and $\sqcap$ such that for all $x, y, z \in D$:

(1) $\perp \sqsubseteq x \sqsubseteq \top$

(2) $x \sqsubseteq x \sqcup y$ and $y \sqsubseteq x \sqcup y$

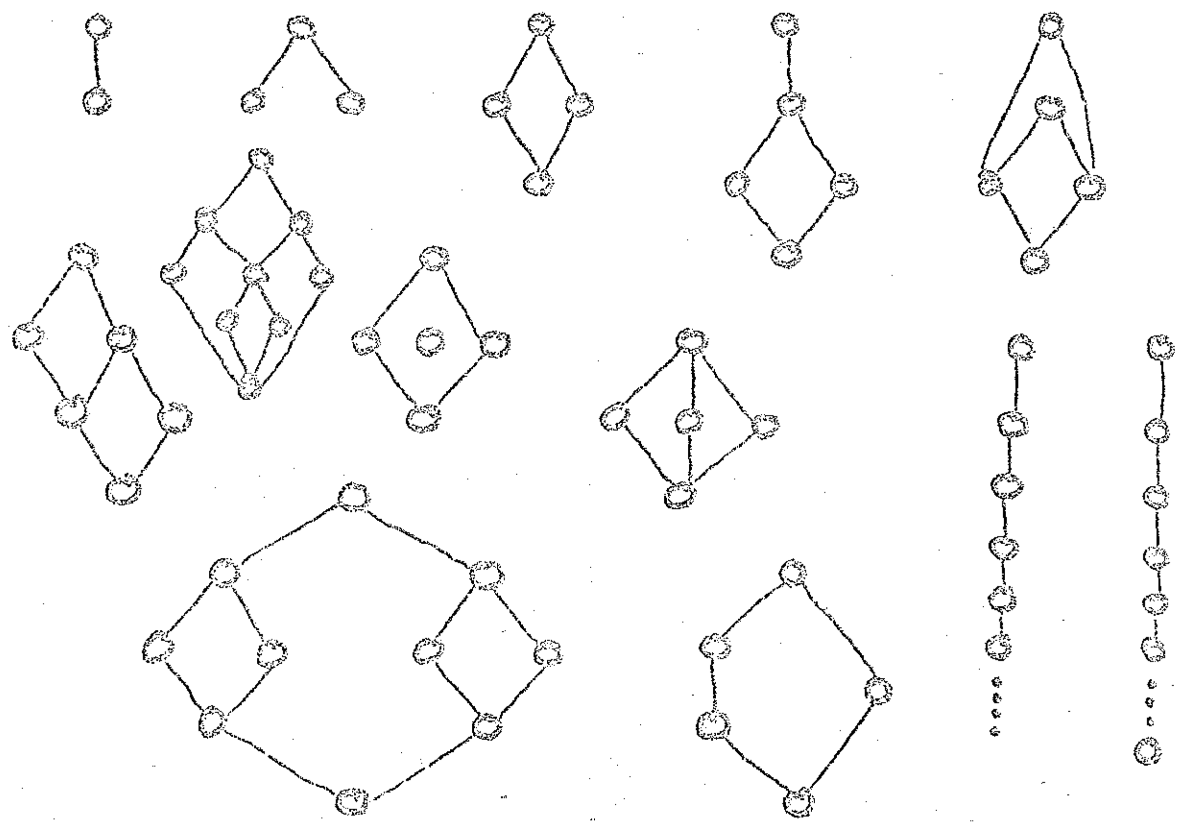
(3) $x \sqsubseteq z$ and $y \sqsubseteq z$ implies $x \sqcup y \sqsubseteq z$

(4) $x \sqcap y \sqsubseteq x$ and $x \sqcap y \sqsubseteq y$

(5) $z \sqsubseteq x$ and $z \sqsubseteq y$ implies $z \sqsubseteq x \sqcap y$

EXERCISE. Show that if $\perp, \top, \sqcup, \sqcap$ exist satisfying (1)-(5), these elements and operations are *unique*.

EXERCISE. Which of these partially ordered sets are lattices:



EXERCISE. Find a partially ordered set which has $\sqcup$ but no $\sqcap$.

EXERCISE. The distributive law reads: $x \sqcap (y \sqcup z) = (x \sqcap y) \sqcup (x \sqcap z)$.
Find non-trivial lattices that satisfy and do not satisfy this law.

EXERCISE. Consider the laws:

(1) $x \sqcup x = x$

(2) $x \sqcup y = y \sqcup x$

(3) $x \sqcup (y \sqcup z) = (x \sqcup y) \sqcup z$

(1') $x \sqcap x = x$

(2') $x \sqcap y = y \sqcap x$

(3') $x \sqcap (y \sqcap z) = (x \sqcap y) \sqcap z$

(4) $(x \sqcap y) \sqcup y = y$

(4') $(x \sqcup y) \sqcap y = y$

(5) $\perp \sqcup x = x$

(5') $\top \sqcap x = x$

(a) Show that every lattice satisfies these ten equations.

(b) Show that any such "algebra" becomes a lattice under the definition:

$$x \sqsubseteq y \text{ iff } x \sqcup y = y.$$

INTUITIVE READINGS.

$x \sqsubseteq y$	x approximates y
$x = \perp$	x is underdetermined or undefined
$x = \top$	x is overdetermined or inconsistent
$z = x \sqcup y$	z is the join of x and y
$z = x \sqcap y$	z is the meet of x and y
$x \sqcup y = \top$	x and y are inconsistent
$x \sqcap y = \perp$	x and y are independent

We call $\perp$ the bottom element and $\top$ the top element. The approximation relation is a qualitative one, and there is nothing implied about the degree of approximation.

FURTHER EXAMPLES. (1) FUN is the partially ordered set of all partial functions from integers to integers together with the extra "inconsistent function" $\top$, where we define:

$f \sqsubseteq g$ iff either $g = \top$ or whenever $f(n)$ is defined then so is $g(n)$ and $f(n) = g(n)$

(2) REL is the partially ordered set of all (binary) relations between integers, where we define:

$R \sqsubseteq S$ iff whenever nRm then nSm.

EXERCISE. Prove that FUN and REL are lattices. What does the equation $f \sqcup g = \top$ mean in FUN? What are $\bot$ and $\top$ in REL?

THE RETRACTION. For $f \in$ FUN and $R \in$ REL we say that

R is the <u>graph</u> of f iff $f \neq \top$ and for all integers n, m $m = f(n)$ iff $n R m$.

Define two mappings i : FUN $\to$ REL and j : REL $\to$ FUN by:

$$i(f) = \begin{cases} R & \text{if } R \text{ is the graph of } f \\ \top & \text{if } f = \top \end{cases}$$

$$j(R) = \begin{cases} f & \text{if } R \text{ is the graph of } f \\ \top & \text{if } R \text{ is not the graph of any function} \end{cases}$$

EXERCISE. Prove that $j(i(f)) = f$ for all $f \in$ FUN and that $i(j(R)) \sqsupseteq R$ for all $R \in$ REL. (This kind of connection between lattices will be met many times.)

AMSTERDAM
JUNE 1972
PROF. SCOTT

CHAPTER II. Recursion.

EXAMPLE. Consider the strict less than relation ($<$) in the (non-negative) integers. It is characterized by these three conditions:

(1) $0 < 1$

(2) $n < m$ implies $n < m+1$

(3) $n < m$ implies $n+1 < m+1$

Certainly these statements are <u>true</u> of $<$, but they also generate <u>all</u> the correct relationships. Let us use $\$$ as a notation for the <u>successor relation</u> and $\breve{\$}$ for the converse, then we have:

$n \mathbin{\$} n'$ iff $n' = n+1$ and $n \mathbin{\breve{\$}} n'$ iff $n = n'+1$.

We can rewrite (1)-(3) as follows:

(1′) $0 < 1$

(2′) $n < m \mathbin{\$} m'$ implies $n < m'$

(3′) $n' \mathbin{\breve{\$}} n < m \mathbin{\$} m'$ implies $n' < m'$

EXERCISE. What are the corresponding conditions for $\leq$? and for $=$?

EXERCISE. Can other relations satisfy (1) - (3)?

DEFINITION. For relations $R, S \in REL$ we define $R;S$ (<u>the relative product</u>) so that:

$$n(R;S)k \text{ iff } nRmSk \text{ for some } m.$$

EXAMPLE. (Cont.) We may rewrite conditions (1′)–(3′) for an arbitrary relation R as follows:

(1″) $0 R 1$

(2″) $R;\mathbb{S} \sqsubseteq R$

(3″) $\breve{\mathbb{S}};R;\mathbb{S} \sqsubseteq R$

Using the notation $\{(0,1)\} \in REL$ for the specific finite relation this becomes:

(1‴) $\{(0,1)\} \sqcup R;\mathbb{S} \sqcup \breve{\mathbb{S}};R;\mathbb{S} \sqsubseteq R$.

According to our intuition, $<$ is the <u>least</u> relation R satisfying (1‴).

EXERCISE. Show that $<$ is in fact the <u>unique</u> relation such that:

$$\{(0,1)\} \sqcup R;\mathbb{S} \sqcup \breve{\mathbb{S}};R;\mathbb{S} = R$$

The next step is to generalize the process of forming recursive definitions illustrated by the example. To this end we introduce the concept of a complete lattice.

DEFINITION. A lattice D is said to be complete iff for all subsets $X \subseteq D$ there exists a (unique) join $\bigsqcup X$ (sometimes called the least upper bound) such that:

(1) $x \sqsubseteq \bigsqcup X$ for all $x \in X$

(2) whenever $x \sqsubseteq z$ for all $x \in X$, then $\bigsqcup X \sqsubseteq z$.

EXERCISE. Every finite lattice is complete.

EXERCISE. The lattices FUN and REL are complete.

EXERCISE. Consider the system of all finite sets of integers and their complements. This is a lattice which is not complete. Find other examples of incomplete lattices

EXERCISE. Consider a group (or semigroup) G. Consider the system $\text{SUB}(G)$ of all subgroups (sub-semigroups) of G. <u>This is a complete lattice</u>. (Hint: any partially ordered set which has arbitrary least upper bounds is a complete lattice. Formulate and prove the same lemma for <u>greatest lower bounds</u>. Which approach is easier to use on $\text{SUB}(G)$?)

EXERCISE. Let A be an arbitrary set. Define an operation $*$ on A such that for all $x, y \in A$:

$$x * y = x.$$

Is $(A, *)$ a semigroup? What is $\text{SUB}(A)$ in this case?

DEFINITION. Let D and D' be lattices (or just partially ordered sets). A function $f: D \to D'$ is called <u>monotonic</u> iff whenever $x \sqsubseteq y$, then $f(x) \sqsubseteq' f(y)$.

Monotonic functions are very important for our study. We shall eventually verify the principle:

All computable functions are monotonic.

Before we can understand this principle, however, we shall have to discuss the precise sense of computability that is appropriate for lattices. This will be done in later chapters. For the time being, we must be content with examples.

EXERCISE. Define a function $F : \mathrm{REL} \rightarrow \mathrm{REL}$ such that:

$$F(R) = \{(0,1)\} \sqcup R ; \$ \sqcup \breve{\$} ; R ; \$.$$

Show that this function is monotonic. Give other examples of monotonic functions on REL. Give an interesting example of a <u>non-monotonic</u> function on REL.

EXERCISE. Let D be any lattice and let $a \in D$ be fixed. The following functions are monotonic:

$$a, \qquad a \sqcup x, \qquad a \sqcap x .$$

EXERCISE. Suppose $f : D \to D'$ and $g : D' \to D''$ are monotonic functions. Define the <u>composition</u> of functions

$$g \circ f : D \to D''$$

by the equation:

$$(g \circ f)(x) = g(f(x)).$$

Show that $g \circ f$ is again monotonic.

EXERCISE. Let $(G, *)$ be a semi-group, and let $A \subseteq G$ be a fixed, non-empty subset. Let $\mathrm{SUB}_0(G)$ be the lattice of all sub<u>sets</u>. Show there is a least subset X such that

$$A \cup A * X \subseteq X$$

where $A * X = \{a * x \,;\, a \in A \text{ and } x \in X\}$. Is $X \in \mathrm{SUB}(G)$ a subsemigroup?

THE FIXED-POINT THEOREM. Let D be a complete lattice, and let $f: D \to D$ be a monotonic function. Then there is a (unique) <u>least</u> element $x \in D$ such that:

$$f(x) \sqsubseteq x .$$

This element is also the least x such that

$$f(x) = x .$$

Proof: We use the notation $\sqcap X$ for greatest lower bounds which must exist in every lattice (Note that $f(\top) \sqsubseteq \top$, so that an x satisfying $f(x) \sqsubseteq x$ need not be unique.)

Let $X = \{y \in D : f(y) \sqsubseteq y\}$, and let $x = \sqcap X$. Whenever $y \in X$, then by definition of $\sqcap$ we have $x \sqsubseteq y$. By monotonicity, $f(x) \sqsubseteq f(y)$. But $y \in X$, so $f(x) \sqsubseteq y$. This holds for all $y \in X$, therefore $f(x) \sqsubseteq \sqcap X = x$. Thus $x \in X$ and $\sqcap X \in X$, which proves the first part of the theorem.

For the second part, notice that by monotonicity $f(f(x)) \sqsubseteq f(x)$. Thus $f(x) \in X$. But then $f(x) \sqsupseteq x$ because $x = \sqcap X$. Therefore $f(x) = x$. If $f(y) = y$, for any y, then $y \in X$ and $x = \sqcap X \sqsubseteq y$. This proves that x is indeed the least fixed point. ∎

NOTE: This proof is highly non-constructive. We shall give a constructive version in the next chapter. This proof does have the advantage that it is very short, however.

EXERCISE. In a complete lattice with at least two elements there always exist monotonic functions that have <u>many</u> fixed points.

EXERCISE. Given $a \in D$, what are the fixed points of the functions $a \sqcup x$ and $a \sqcap x$.

EXERCISE. In a semigroup $(G, *)$ discuss the functions $A \cup A * X$ and $A \cup X * X$, where $X \in \text{SUB}_0(G)$.

AMSTERDAM
JUNE 1972
PROF SCOTT

CHAPTER III. Continuity

But if addiction to philosophical theories is like an illness, it is a necessary illness, because, without it, the empirical investigation of language would lose its point. "Wittgenstein" D. Pears.

In Chapter II we proved the fixed-point theorem by a very general, but non-constructive method. In particular, that proof gives very little immediate information about the computability of the objects defined as such fixed points. The purpose of discussing the notion of continuous function is to make the existence proof more definite. Indeed as part of our dogma we shall assert:

All computable functions are continuous.

This will be justified on the basis of a very rough idea of computability.

AN INTUITIVE ARGUMENT. Suppose $f: D \to D'$ is computable (in some reasonable sense). In these lattices the approximation relationship $x \sqsubseteq y$ is supposed to mean that the "information content" of x is included in that of y (i.e. y has "more" and "better" information). If there is to be some notion of computability, information will have to be determined by "finite" approximation. And a function will only be able to compute in terms of these finite approximations. More precisely, if $f(x)$ is computable, we should be able to say that a finite amount of information about $f(x)$ is already determined by a finite amount of information about x — otherwise there would not be any effective way of doing the computation.

The only trouble with this argument is that the idea of "finite approximation" has not been made precise. Fortunately there is a very simple way of formulating the conclusion without making a definition of "finite".

Suppose elements $x_n \in D$, $n=0,1,2,\ldots$, are given so that:

$$x_0 \sqsubseteq x_1 \sqsubseteq x_2 \sqsubseteq \cdots \sqsubseteq x_n \sqsubseteq x_{n+1} \sqsubseteq \cdots$$

Information here is increasing monotonically. Since the lattice is complete, we can form

$$\bigsqcup_{n=0}^{\infty} x_n = \bigsqcup \{x_n : n=0,1,2,\ldots\}$$

In this case the least upper bound is very naturally called a <u>limit</u> and it is a limit with <u>infinitely many</u> terms. THUS it seems reasonable to suppose that any finite amount of information about the limit is already contained in <u>one</u> of the terms.

Suppose then that $f: D \to D'$ is computable, and hence, monotonic. Consider f evaluated at a limit. In view of the foregoing remarks we ought to be able to write

$$f\left(\bigsqcup_{n=0}^{\infty} x_n\right) = \bigsqcup_{n=0}^{\infty} f(x_n)$$

At least the finite amounts of information are the same on both sides — and we shall assume an element is determined by its finite information content. This leads us to the formal definitions.

DEFINITION. A subset $X \subseteq D$ of a lattice is called directed iff X contains an upper bound for each of its finite subsets.

REMARK. A directed set is always non-empty. The definition generalizes the notion of a chain, $\{x_n : n = 0,1,2,\ldots\}$ where $x_n \sqsubseteq x_{n+1}$ for all n. The least upper bound $\bigsqcup X$ can be called the limit of the directed set.

DEFINITION. A function $f: D \to D'$ is called continuous iff it preserves limits; that is, for every directed $X \subseteq D$:

$$f(\bigsqcup X) = \bigsqcup' \{f(x) : x \in X\}.$$

EXERCISE. Continuous functions are monotonic. On a finite lattice the converse holds, but this is not so in general.

EXERCISE. Find examples of continuous functions on REL and SUB(G) from Chapt. II.

THE CONTINUOUS FIXED-POINT THEOREM. Let $f: D \to D$ be continuous. Then the least fixed point $x = f(x)$ is given by the iteration formula:

$$x = \bigsqcup_{n=0}^{\infty} f^n(\bot),$$

where $f^n(\bot) = \underbrace{f(f(f \cdots f}_{n\text{-times}}(\bot)\cdot))$.

Proof: Let $x = f(x)$ be the least fixed point. Now $\bot \sqsubseteq x$, so $f(\bot) \sqsubseteq f(x) = x$, by monotonicity of f. Clearly by iteration $f^n(\bot) \sqsubseteq x$ for all n. Thus the limit is also contained in x. Next note that the first term $f^0(\bot) = \bot$ of the limit is superfluous. Hence:

$$f(\bigsqcup_{n=0}^{\infty} f^n(\bot)) = \bigsqcup_{n=0}^{\infty} f^{n+1}(\bot) = \bigsqcup_{n=0}^{\infty} f^n(\bot)$$

by the continuity of f and the fact that $\bot \sqsubseteq f(\bot) \sqsubseteq f^2(\bot) \sqsubseteq \cdots \sqsubseteq f^n(\bot) \sqsubseteq f^{n+1}(\bot) \sqsubseteq \ldots$. So the limit <u>is</u> a fixed point and therefore must contain x.

EXERCISE. Check the formula for the examples of Chapter II.

EXAMPLE. The natural and constructive character of the iteration formula can be appreciated in a discussion of integer recursion. Consider the recursive definition :

$$f(n) = \text{if } n=0 \text{ then } 0 \text{ else } f(n-1)+2n-1$$

As is well known this defines the function $f(n) = n^2$, a total function. This total function can be approximated by partial functions in FUN. We define

$$F : \text{FUN} \to \text{FUN}$$

$$F(g) = \begin{cases} \top & \text{if } g = \top \\ & \text{otherwise} \\ \lambda n.\ \text{if } n=0 \text{ then } 0 \text{ else } g(n-1)+2n-1, \end{cases}$$

where λ-abstraction on an integer variable has been used to specify a function. EXERCISE: Check that F is continuous and that indeed

$$f = \bigsqcup_{n=0}^{\infty} F^n(\perp).$$

(Hint: make a table of the partial functions $F^n(\perp)$.)

EXERCISE. Show that the composition of continuous functions is continuous.

EXERCISE. Suppose that $f, g : D \to D$ are two continuous functions such that

$$f \circ g = g \circ f.$$

Show that f and g have a least <u>common</u> fixed point $x = f(x) = g(x)$. Find the formula for x. Is this x necessarily the least fixed point of f or of g? Is it a fixed point of $f \circ g$?

We shall often want to talk of functions of several variables; the use of Cartesian products makes this very easy.

DEFINITION. Given lattices D, D' we define $D \times D'$ to be the set of pairs (x, x') where $x \in D$ and $x' \in D'$ with this relation:

$$(x, x') \sqsubseteq (y, y') \text{ iff } x \sqsubseteq y \text{ and } x' \sqsubseteq' y'.$$

EXERCISE. $D \times D'$ is obviously a partially ordered set. Prove that it is a lattice (if D and D' are) and that it is complete (if D and D' are).

EXERCISE. Let D and D' be complete lattices. Show that the following functions are indeed continuous:

$$
\begin{aligned}
first &: D \times D' \to D \\
second &: D \times D' \to D' \\
twist &: D \times D' \to D' \times D \\
merge &: (D \times D') \times (D \times D') \to D \times D' \\
join &: D \times D \to D \quad , \\
in &\quad D \to D \times D
\end{aligned}
$$

where:

$$
\begin{aligned}
first(x, x') &= x \\
second(x, x') &= x' \\
twist(x, x') &= (x', x) \\
merge((x, x'), (y, y')) &= (x, y') \\
join(x, x') &= x \sqcup x' \\
in(x) &= (x, x)
\end{aligned}
$$

DEFINITION. A pair of (continuous) functions $i : D \to D'$ and $j : D' \to D$ is called a <u>retraction</u> iff for all $x \in D$:

$$j(i(x)) = x .$$

EXERCISE. Find additional functions so that each of the five functions in the last Exercise becomes a retraction.

DEFINITION. A retraction $i : D \to D'$ and $j : D' \to D$ is called a <u>projection</u> iff in addition we have for all $x' \in D'$:

$$i(j(x')) \sqsubseteq x'$$

EXERCISE. Which of the five functions are projections? What can be said for the function $\mathrm{meet} : D \times D \to D$, where

$$\mathrm{meet}(x, x') = x \sqcap x' \quad ?$$

EXERCISE. Let D, D', and D'' be complete lattices. Prove that a function

$$f : D \times D' \to D''$$

is continuous iff for all $x \in D$ and for all $x' \in D'$ each of these functions is continuous:

$$\lambda t'.\, f(x, t') : D' \to D''$$
$$\lambda t.\, f(t, x') : D \to D''$$

(A function of two variables is jointly continuous iff it is continuous in each of the variables separately.)

EXERCISE. Define cartesian powers D^n (more abstractly D^I, where I is any set). Discuss completeness and projections. Let $\mathbb{O} = \{\bot, \top\}$ be the two element lattice. Identify in familiar terms the powers $\mathbb{O}^I$ and $\mathbb{O}^{I \times J}$, where $I \times J$ is just the cartesian product of sets.

DEFINITION. A (continuous) function $j : D \to D$ from a (complete) lattice into itself is said to be a <u>retraction</u> iff $j \circ j = j$. It is a <u>projection</u> iff in addition $j(x) \sqsubseteq x$ for all $x \in D$.

EXERCISE. If $j : D \to D$ is a retraction, then the range of values of j is the same as the set of fixed points of j.

EXERCISE. Let $f : D \to D$ be monotonic and D be complete. Show that the set of fixed points of f:

$$\mathrm{FIX}(f) = \{x \in D : x = f(x)\}$$

is itself a complete lattice. (<u>Hint</u>:

Suppose $X \subseteq \mathrm{Fix}(f)$. Define:

$$\bar{x} = \bigsqcap \{z : f(z) \sqsubseteq z,\ \bigsqcup X \sqsubseteq z\}$$

and try to show $\bar{x} \in \mathrm{Fix}(f)$.) Corollary: the range of a retraction is a complete lattice.

EXERCISE. Suppose $i : D \to D'$ and $j : D' \to D$ is a retraction pair. What can be said about

$$j' = j \circ i : D' \to D'$$

and about its range?

EXERCISE. Suppose $j : D' \to D'$ is a projection. Let D be the range of j so that we can regard $j : D' \to D$. Define $i : D \to D'$ by the restriction of the identity function. This is a projection pair. Show that j can be defined in terms of D by the formula

$$j(x') = \bigsqcup \{x \in D : x \sqsubseteq x'\}$$

where the $\bigsqcup$ is taken in D.

Suppose $X \subseteq \mathrm{Fix}(f)$. Define:

$$\bar{x} = \bigsqcap \{z : f(z) \sqsubseteq z,\ \bigsqcup X \sqsubseteq z\}$$

and try to show $\bar{x} \in \mathrm{FIX}(f)$.) Corollary: the range of a retraction is a complete lattice.

EXERCISE. Suppose $i : D \to D'$ and $j : D' \to D$ is a retraction pair. What can be said about

$$j' = j \circ i : D' \to D'$$

and about its range?

EXERCISE. Suppose $j : D' \to D'$ is a projection. Let D be the range of j so that we can regard $j : D' \to D$. Define $i : D \to D'$ by the restriction of the identity function. This is a projection pair. Show that j can be defined in terms of D by the formula

$$j(x') = \bigsqcup \{x \in D : x \sqsubseteq x'\}$$

where the $\bigsqcup$ is taken in D.

REMARK. More information on projections and retractions is given in the paper "Continuous Lattices".

The notion of cartesian product makes it very easy to talk about <u>simultaneous equations</u> Thus suppose D is a complete lattice and that

$$f : D \times D \to D$$
$$g : D \times D \to D$$

are continuous (or simply: monotonic). Then we can ask for solutions to the pair of equations:

$$x = f(x, y)$$
$$y = g(x, y) \,.$$

A minimal <u>pair</u> can be shown to exist by defining

$$F : D \times D \to D \times D$$

by the formula

$$F(x, y) = (f(x, y), g(x, y)) \,.$$

Now we know that a minimal fixed point exists in $D \times D$ so that

$$(x, y) = F(x, y) = (f(x, y), g(x, y))$$

This gives the desired pair of equations.

EXERCISE. In the solution just given, will it be the case that in the single equation

$$x = f(x, y)$$

the x we found is a minimal solution? (This may not be so easy to answer at this stage.) Is there any simple formula for the pair (x, y)? (This may not be so easy either.)

EXERCISE. Let $(G, *)$ be the free semigroup (with identity if you like) on the "alphabet" $A = \{u, v, \to, =, \vee, \neg, (,), ;\}$. Consider the lattice $\mathrm{SUB}_0(G)$ of all subsets of G. Discuss the simultaneous solution of this pair of equations:

$$X = \{u, v\} \cup \{(\} * Y * \{\to\} * X * \{;\} * X * \{)\}$$

$$Y = \{(\} * X * \{=\} * X * \{)\} \cup \{\neg\} * Y \cup$$

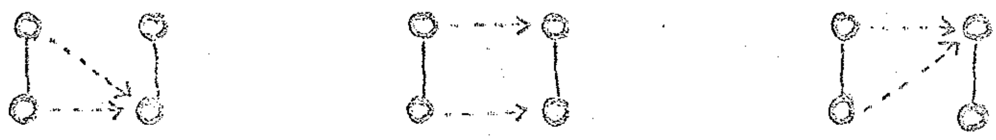
$$\{(\} * Y * \{\vee\} * Y * \{)\}.$$

CHAPTER IV. Topology

If on some space or the other it is reasonable to define a class of functions which can be called <u>continuous</u>, then there must be a <u>topology</u> somewhere in the background. In our case we have as spaces the complete lattices, and the notion of continuous function has been defined in the last chapter. The discovery of the right topology on the lattices will be easy.

There is in the first place no problem with the one-element lattice: there are two open sets, the empty set and the whole set, and the open sets of a space (by definition) determine its topology - in this case it is trivial. Consider next the two-element lattice illustrated in the figure. (It is called $\mathbb{O}$ because it has <u>NO</u> interesting elements.) On this lattice we can define only three continuous functions into $\mathbb{O}$ itself. They are shown thus:

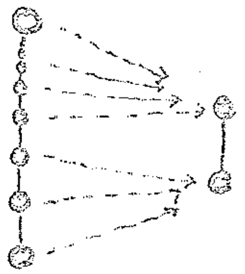
THE LATTICE $\mathbb{O}$



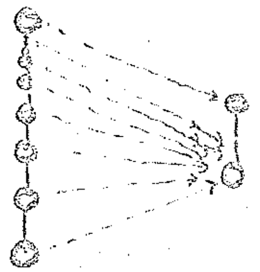
The one function that is being excluded is the function that interchanges $\perp$ and $\top$. Now a topological definition of continuous function is that the function should preserve openness of sets under inverse images. That is, if $f: D \to D'$ is continuous and $U' \subseteq D'$ is open, then

$$\{x \in D : f(x) \in U'\}$$

must be open too. In the case of $\mathbb{O}$ we shall need more than only the open sets $\emptyset$ and $\{\perp, \top\}$, since otherwise no functions would be excluded as discontinuous. There are only two other subsets $\{\perp\}$ and $\{\top\}$, and the question of which is open cannot be decided by the above class of functions because of the obvious symmetry in the situation. An infinite lattice is required as follows:

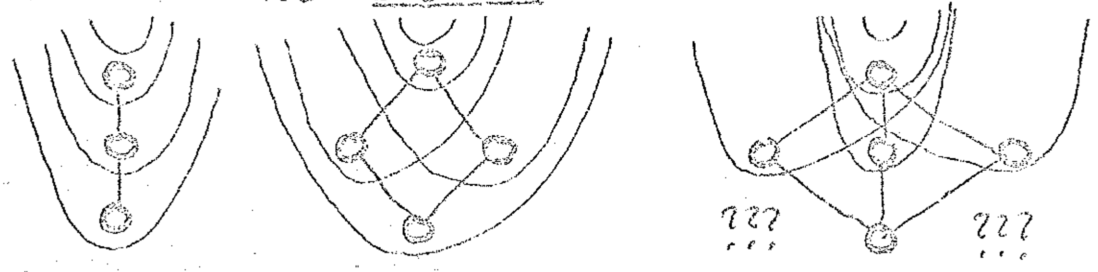


CONTINUOUS



DISCONTINUOUS

The lattice has only one proper limit point (there can be no proper limit points in finite lattices), and the function on the right is discontinuous. The reason is that all but the limit point are mapped to $\perp$ while it goes to $\top$. If the function were continuous, $\top$ would have to go to $\perp$ by the definition on p. 20. In the example on the left, the first few terms of the limit go to $\perp$, but the rest go to $\top$. This function is continuous. If $\{\perp\}$ were open, then any finite lower section of the infinite lattice would be open. But then the <u>union</u> of all those open sets would be open; that is, the set of all elements except the limit point. But then the discontinuous function would be continuous. It cannot be so, therefore $\{\perp\}$ is <u>not</u> open. Rather it is $\{\top\}$ that is the open subset of $\mathbb{O}$; while $\{\bot\}$ is closed.



$\mathbb{1}$ $\mathbb{2}$ $\mathbb{3}$

EXERCISE. The lattices $\mathbb{1}$, $\mathbb{2}$, and $\mathbb{3}$ are illustrated $\mathbb{1}$ has 4 open sets (as shown) and $\mathbb{2}$ has 5. Refine the argument for $\mathbb{O}$ to prove this. How many open sets does $\mathbb{3}$ have? (They could not be all shown in the figure.) What can you say about other finite lattices?

DEFINITION. A subset of a complete lattice, say $U \subseteq D$, is said to be open iff the following two conditions are satisfied:

(i) whenever $x \in U$ and $x \sqsubseteq y$, then $y \in U$.

(ii) whenever $S \subseteq D$ is directed and $\bigsqcup S \in U$, then $x \in U$ for some $x \in S$.

These two conditions are <u>necessary</u>, because if $U \subseteq D$ is to be open, then, in view of the topology of $\mathbb{O}$, the function $f : D \to \mathbb{O}$ such that

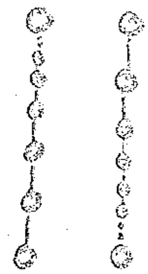
$$f(x) = \begin{cases} \top & \text{if } x \in U \\ \bot & \text{if } x \notin U \end{cases}$$

has to be continuous. If it is, then from the equation in the definition on p. 20 the set U has to satisfy (i) and (ii). <u>Conversely</u>, if U satisfies (i) and (ii), then f is continuous. This means that U must be open. Hence, the choice of topology is completely determined. (EXERCISE: Check these assertions.)

EXERCISE. Write $U \in \text{OPEN}(D)$ to mean that U is open. Show from the definition that the family of subsets $\text{OPEN}(D)$ is closed under arbitrary unions and finite intersections. What about infinite intersections?

EXERCISE. Show that $\text{OPEN}(D)$ is a complete lattice.

EXERCISE. What are the open subsets of the two infinite lattices shown in the diagram?



EXERCISE. Let D be a complete lattice and let $y \in D$. Show that

$$\{x \in D : x \not\sqsubseteq y\} \in \text{OPEN}(D).$$

EXERCISE. Prove that in any complete lattice D and for any $x, y \in D$ we have

$x \sqsubseteq y$ iff whenever $x \in U \in \text{OPEN}(D)$, then $y \in U$.

DEFINITION. Let X be any topological space and $\text{OPEN}(X)$ be its class of open subsets. The space X is said to be a T_0-*space* iff for all $x, y \in X$:

$$x = y \text{ iff } \{U : x \in U\} = \{U : y \in U\},$$

where the set variable U ranges over $\text{OPEN}(X)$. ("Points are uniquely determined by their neighborhoods.")

EXERCISE. Every complete lattice is a T_0-space.

In the paper "Continuous Lattices" it is shown that every T_0-space can be embedded in a continuous lattice. (We define what a continuous lattice is below.) The proof is well known from topology. Take any T_0-space X and let $\mathcal{G} = \mathrm{OPEN}(X)$. Regard $\mathcal{G}$ simply as a set. Let $D = \mathbb{O}^{\mathcal{G}}$ be the lattice formed as the cartesian power of $\mathbb{O}$. The "points" of D correspond to classes of open subsets of X. Define the embedding $e: X \to D$ by the formula:

$$e(x) = \{U : x \in U\}.$$

It is easy to check that e is one-one, continuous, and is actually a homeomorphism (topological isomorphism) of X with a topological subspace of D. This is one reason why in this study it is useful to work with lattices.

EXERCISE. Let D and D' be two complete lattices. Show that a function $f : D \to D'$ is contenuous in the ordinary topological sense (inverse images of open sets are open) iff it satisfies the lattice-theoretic definition.

DEFINITION. An element $x \in D$ is said to be <u>isolated</u> iff the set

$$\{y \in D : x \sqsubseteq y\} \in \mathrm{OPEN}(D).$$

EXERCISE. In a finite lattice <u>all</u> elements are isolated. What are the isolated points in the infinite lattices shown on p. 35? Find a complete lattice of at least two elements where the only isolated point is $\perp$.

EXERCISE. $x \in D$ is isolated iff whenever $S \subseteq D$ is directed and $x \sqsubseteq \bigsqcup S$, then $x \sqsubseteq y$ for some $y \in S$ ("Isolated points cannot be reached in a non-trivial way by limits.)

EXERCISE. Find some infinite lattices in which all points are isolated. In such lattices D, a function $f: D \to D'$ into an arbitrary lattice is continuous iff it is monotonic.

EXERCISE. What are the isolated points of FUN, REL, SUB(G)?

Isolated points might well be called finite points, because they cannot be reached by limits and somehow stand alone. Of course this is a very generalized notion of finite and is only an analogy. The idea, however, is a good one and leads to a relativized definition:

DEFINITION. In a complete lattice D an element x is said to be relatively finite with respect to an element y, in symbols: $x \prec y$, iff whenever $S \subseteq D$ is directed and $y \sqsubseteq \bigsqcup S$, then $x \sqsubseteq z$ for some $z \in S$.

Clearly isolated points are those such that $x \prec x$ holds. The relation $\prec$ has many nice properties.

EXERCISE. In a complete lattice prove that

(i) $x \prec y$ implies $x \sqsubseteq y$;

(ii) $x \sqsubseteq y \prec z \sqsubseteq w$ implies $x \prec w$;

(iii) $\prec$ is transitive;

(iv) $\perp \prec x$;

(v) $x \prec y$ and $x' \prec y$ implies $x \sqcup x' \prec y$.

EXERCISE. Consider the lattice $\mathbb{I}$ which consists of the real numbers in the closed interval $[0,1]$ where $\sqsubseteq$ means $\leq$. (Draw it vertical rather than horizontal.) Show that $\prec$ means $<$ (except at $\perp = 0$).

In "Lattice Theory, Data Types, and Semantics", the lattice R of interval analysis is described (p. 83). (Note that on p. 84 the figure for the limit is wrong. Why?) The properties of this lattice should be worked out as an exercise.

In a lattice which has non-isolated points we can roughly interpret the relationship $x \prec y$ as meaning that x is a "finite" approximation to y. In lattices such as FUN, REL, and SUB(G), this is a very good reading. These lattices also have the special property that every element *is* the limit of its finite approximations. This is a very useful property which is possessed by a wide range of lattices. We make it a formal definition.

DEFINITION. A complete lattice D is said to be a *continuous* lattice iff for all $y \in D$ we have:

$$y = \bigsqcup \{x \in D : x \prec y\}.$$

The reason for the word "continuous" is first that approximations in such lattices have a pleasant sort of continuous quality; and then in these lattices so many functions *are* continuous.

EXERCISE. Let D be a continuous lattice. Show that for $y \in D$:

$$y = \bigsqcup\{\sqcap U : y \in U\}$$

where U ranges over OPEN(D). (Hint: Show first that

$$\{y : x \prec y\} = \bigcup\{U : U \subseteq \{z : x \sqsubseteq z\}\}.)$$

(See "Continuous Lattices" p. 6.)

EXERCISE. Let D be a continuous lattice, then for $x, y \in D$

$$x \sqsubseteq y \text{ iff whenever } z \prec x, \text{ then } z \sqsubseteq y.$$

EXERCISE. Let D, D' be continuous lattices. Then $f : D \to D'$ is a continuous function iff whenever $z \prec f(y)$, then $z \sqsubseteq f(x)$ for some $x \prec y$.

EXERCISE. Use the previous exercise to show that in any continuous lattice the meet function $\sqcap$ is continuous. (Cf. "Continuous Lattices" p. 9.)

The main reason for using continuous lattices (and indeed lattices) is the possibility of having so many continuous functions. This feature will become especially obvious in the next chapter on function spaces. In continuous lattices we also find that retracts work out especially well. This will be discussed in the chapter on infinite types. In the paper "Continuous Lattices" it is shown that a continuous lattice (as a topological space) is just one such that it is a retract of every space of which it is a subspace. This is a consequence of the important:

THE EXTENSION THEOREM. Let D be a continuous lattice. Let X and Y be any two topological spaces such that $X \subseteq Y$ as a subspace. Suppose $f: X \to D$ is continuous. Then f can be extended to a continuous function $\bar{f}: Y \to D$. This extension property is in fact characteristic of continuous lattices.

We shall not give the proof here in full, but the idea will be explained. The function $\bar{f}$ can be defined by this formula:

$$\bar{f}(y) = \bigsqcup\{\bigsqcap\{f(x) : x \in X \cap U\} : y \in U\}$$

where U ranges over OPEN(Y). That is, in each neighborhood of y we look at the "total variation" of the function f on the portion of U on which f is well defined. We then take the limit of these variations as the neighborhoods converge to y. This function $\bar{f}$ would be well defined in any complete lattice, but we need the assumption that D is a continuous lattice in order for $\bar{f}$ to be continuous and for $\bar{f}(x) = f(x)$ for all $x \in X$.

This method is exactly the way that (partial) continuous functions are made into interval functions in the lattice R. (Cf. "LT, DT, Sch" pp. 85-89.)

CHAPTER V. Function Spaces.

The real reason for introducing all this theory is to be able to pass to the higher types. The recursion process only gains real power when it can be applied to higher-type operators. In order to make this precise, we need to be able to regard function spaces as lattices. Then it will be possible to treat operators (mappings between function spaces) by the same theory we applied to functions on lattices.

DEFINITION. Given complete lattices D and D', let $[D \to D']$ be the set of all continuous functions $f: D \to D'$ where a partial ordering is defined by

$$f \sqsubseteq g \quad \text{iff} \quad f(x) \sqsubseteq' g(x) \text{ for all } x \in D.$$

THEOREM. If D and D' are complete (continuous) lattices, then so is $[D \to D']$.

Proof: The completeness is easy to establish because we need only show the existence of least upper bounds. Suppose $\mathcal{F} \subseteq [D \to D']$ is a set of continuous functions. Then define:

$$(\bigsqcup \mathcal{F})(x) = \bigsqcup \{f(x) : f \in \mathcal{F}\}.$$

It is straight foward to verify that $\bigsqcup \mathcal{F}$ is continuous, and then it is obvious that it is the l.u.b..

In case D and D' are continuous lattices, it is more difficult to prove that $[D \to D']$ is continuous. The full proof is given in "Continuous Lattices" where it is shown that every continuous function can be approximated by <u>step functions</u>. A step function is a finite join of elementary functions $\vec{e}(e, e')$ where

$$\vec{e}(e,e') = \begin{cases} e' & \text{if } e \prec x \\ \bot & \text{otherwise} \end{cases}$$

These functions are obviously continuous in view of the properties of $\prec$. What can be shown for approximations is:

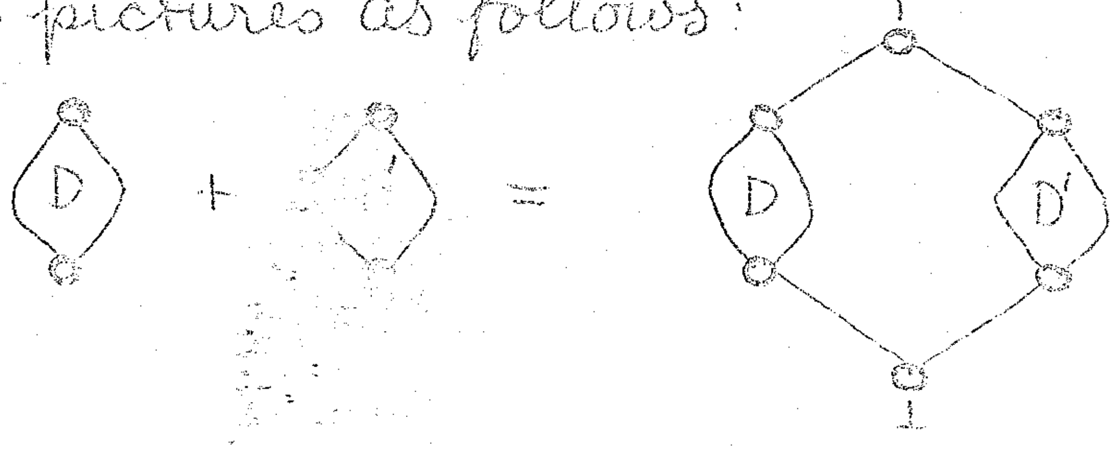
$$f = \bigsqcup \{ \vec{e}(e,e') : e' \prec f(e) \}$$

and every such $\vec{e}(e,e') \prec f$. Hence for all $f \in [D \to D']$ we would have

$$f = \bigsqcup \{ g : g \prec f \},$$

and this means that the lattice $[D \to D']$ is continuous. $\square$

REMARK. Not only does the construction $[D \to D']$ produce continuous lattices, but so do $[D \times D']$ and $[D + D']$. The latter operation on lattices is defined in pictures as follows:



Thus $[D+D']$ is a disjoint union with a new $\perp$ and $\top$ added to make the partially ordered set into a lattice.* By combining these operations, we get a very rich collection of lattices. What makes them interesting is the variety of continuous functions that can be defined on them. Also important are the many retracts that are available. A further operation (inverse limit of lattices) will be discussed in the next chapter.

EXERCISE. Verify that $[D\times D']$ and $[D+D']$ <u>are</u> continuous lattices if D and D' are. (To work this problem, give a precise set-theoretical construction of the set $[D+D']$ with its partial ordering as suggested by the pictures.)

* This construction is slightly different from the + used in other papers, but John Reynolds convinces me that the one used here is better.

THEOREM. Given complete lattices D, D', D'', the mappings:

$$\mathrm{apply}: [D \to D'] \times D \to D'$$
$$\mathrm{abstract}: [D \times D' \to D''] \to [D \to [D' \to D'']]$$

are continuous where we define:

$$\mathrm{apply}(f, x) = f(x)$$
$$\mathrm{abstract}(f)(x)(y) = f(x, y).$$

Proof: To check that apply as a function of two variables is continuous, we need check each variable separately. Now if you fix f, then $f(x)$ is clearly continuous in x because f is. Next fix x. The formula for $\bigsqcup \mathcal{F}$ on p. 45 shows why $f(x)$ is continuous in f.

For abstract it is easier to think in terms of the λ-<u>notation</u>. We would write

$$\mathrm{abstract}(f) = \lambda x.\, \lambda y.\, f(x, y)$$

where x ranges in D and y in D'. Now for each x, we know that $\lambda y. f(x,y)$ is in $[D' \to D'']$, because $f(x,y)$ is continuous in y. BUT the question now is whether the expression $\lambda y. f(x,y)$ is continuous in x. The formula on p. 45 can be written in λ-notation as:

$$\bigsqcup_{i \in I} f_i = \lambda y. \left[\bigsqcup_{i \in I} f_i(y)\right]$$

where $\{f_i : i \in I\}$ is any indexed set of functions. Thus if $X \subseteq D$ is directed, we could use this formula (from right to left) to write:

$$\lambda y. f(\bigsqcup X, y) = \lambda y. \bigsqcup_{x \in X} f(x,y)$$

$$= \bigsqcup_{x \in X} \lambda y. f(x,y).$$

This establishes the continuity in x. Thus $\lambda x. \lambda y. f(x,y)$ is in $[D \to [D' \to D'']]$. Now by the same argument we can show that $\lambda x. \lambda y. f(x,y)$ is continuous in f. $\square$

EXERCISE. Establish the continuity of the (typed!!) combinators:

$$I : D \to D$$

$$K : D \to [D' \to D]$$

$$S : [D \to [D' \to D'']] \to [[D \to D'] \to [D \to D'']]$$

where

$$I = \lambda x.\, x$$

$$K = \lambda x.\, \lambda y.\, x$$

$$S = \lambda f.\, \lambda g.\, \lambda x.\, f(x)(g(x))$$

Can you make any sense of the equation

$$S(K)(K) = I$$

(Hint: The two K's may not be the same.)

EXERCISE. Let $\omega = \{0, 1, 2, \ldots\}$ be the set of integers, and let D^ω be the infinite cartesian power of the complete lattice D. Define

$$\bigsqcup : D^\omega \to D$$

by

$$\bigsqcup(s) = \bigsqcup_{n=0}^{\infty} s_n$$

Show that $\bigsqcup$ is continuous.

THEOREM. The fixed-point operator $Y: [D \to D] \to D$ is continuous.

Proof: By the Continuous Fixed-Point Theorem we can write

$$Y(f) = \bigsqcup_{n=0}^{\infty} f^n(\bot)$$

But now each $f^n(\bot)$ is continuous in f, so therefore must the $\bigsqcup$ be also.

DISCUSSION. The answer to the question on simultaneous equations on p. 29 can now be explained. By using functions from $D \times D$ to $D \times D$ we found the minimal pair that solves:

$$x_0 = f(x_0, y_0)$$
$$y_0 = g(x_0, y_0)$$

The question was (with y_0 fixed) whether x_0 is also the minimal solution of:

$$x_0 = f(x_0, y_0).$$

This will be established by proving that: $x_0 = Y(\lambda x.\, f(x, y_0))$

$$y_0 = Y(\lambda y.\, g(Y(\lambda x. f(x,y)), y))$$

Since we do not know that these elements do form the minimal pair, call them x_1, y_1 first. By the fixed-point property we will have

$$y_1 = g(Y(\lambda x.\, f(x, y_1)), y_1)$$

and so

$$y_1 = g(x_1, y_1)$$

Also

$$x_1 = f(x_1, y_1).$$

Thus (x_1, y_1) is a fixed-point pair, which proves that $x_0 \sqsubseteq x_1$ and $y_0 \sqsubseteq y_1$. We want to establish the full equality.

Let $x_2 = Y(\lambda x.\, f(x, y_0))$. (By monotonicity, since $y_0 \sqsubseteq y_1$, we have $x_2 \sqsubseteq x_1$. a mistake! this is not relevant.) By the fixed-point properties $x_2 \sqsubseteq x_0$. Thus

$$g(Y(\lambda x.\, f(x, y_0)), y_0) \sqsubseteq g(x_0, y_0) = y_0$$

Now in view of the fixed-point operator theorem on p. 15 we see $y_1 \sqsubseteq y_0$. Thus $y_1 = y_0$. But then:

$$f(x_0, y_1) = f(x_0, y_0) = x_0$$

and therefore $x_1 \sqsubseteq x_0$, which proves equality.

This kind of use of the iterated fixed-point operator has been exploited by de Bakker in "Recursive Procedures" (MC Tract #24) where extensive references are given.

EXERCISE. Show that the function

$$K : D \to [D \to D]$$

is one term of a retraction (actually: projection) where the partial inverse is:

$$M : [D \to D] \to D$$

such that $M(f) = f(\perp)$. Can there be other retractions of $[D \to D]$ onto D ???

THEOREM. Let D be a complete lattice. The retracts of D onto itself themselves form a complete lattice as a sub-partial ordering of $[D \to D]$.

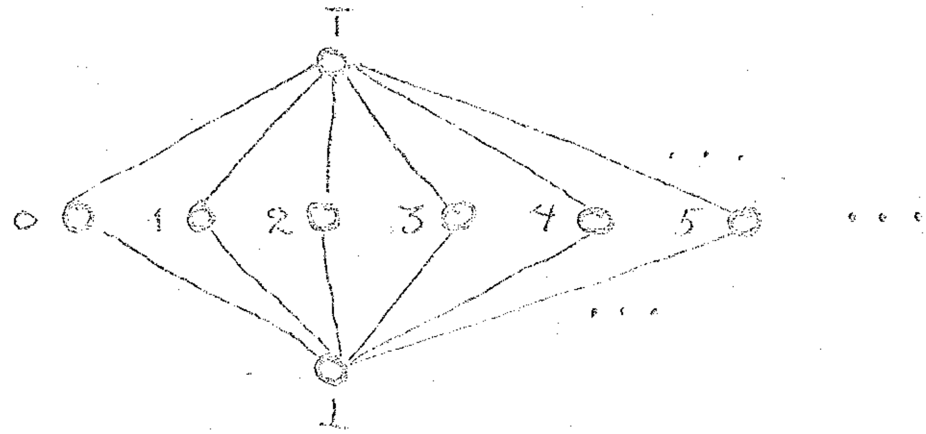
Proof: Let $Q: [D \to D] \to [D \to D]$ be defined by

$$Q(f) = f \circ f = \lambda x. f(f(x)).$$

Now Q is a continuous function and its fixed points are exactly the retracts. But the fixed points of <u>any</u> continuous function form a complete lattice. $\square$

REMARK. The fixed points may not form a complete sublattice since they need not be closed under $\sqcup$. However they are closed under directed limits. Furthermore, the retracts also contain $\perp = \lambda x. \perp$. Therefore fixed-point constructions by operators that map retracts to retracts will yield retracts. We shall employ this important observation many times.

The function-space construction combined with retracts can be regarded as a way of regularizing many other constructions. We illustrate this in the following exercises, where ω is the set of integers and N is the lattice:



EXERCISE. Show that FUN is a retract of N^ω which is in turn a retract of $[N \to N]$.

EXERCISE. Recall that $\mathbb{O}$ is the $\{\bot, \top\}$-lattice. Now REL is the lattice $\mathbb{O}^{\omega \times \omega}$, but show it is a retract of $[N \times N \to \mathbb{O}]$. Define on this latter space the analogues of the usual operations on relations – the continuous ones that is!

EXERCISE. Let D_0 be a retract of D_0' and D_1 be a retract of D_1'. Show that the same relationship holds between:

$$D_0 \times D_1 \quad \text{and} \quad D_0' \times D_1'$$

$$D_0 + D_1 \quad \text{and} \quad D_0' + D_1'$$

$$D_0 \to D_1 \quad \text{and} \quad D_0' \to D_1'$$

(Hint: the last result is most easily proved by reference to the diagram:

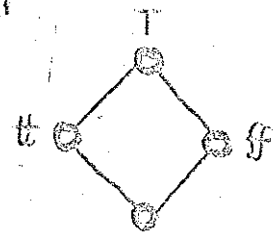
$$\begin{tikzcd} D_0 \arrow[r, shift left] \arrow[d] & D_0' \arrow[l, shift left] \arrow[d] \\ D_1 \arrow[r, shift left] & D_1' \arrow[l, shift left] \end{tikzcd}$$
.)

CHAPTER VI. Infinite Types

The higher types introduced in the last chapter (combinations of $D \times D'$, $D + D'$, and $D \to D'$) are very useful for indicating in a simple and precise way the logical characteristics of large numbers of functions and operators. It is by means of such functions that we specify structure. However, there is the problem of proving the <u>existence</u> of suitably structured domains, and the simple combinations of higher types may not be adequate for this purpose. This is one of the main reasons for introducing the infinite-type spaces, the other reason being that the calculus of functions on the infinite-type space is a neat yet powerful mode of expression.

There are many infinite-type spaces that could be defined, but we shall find that one of them is more than sufficient, because so many others can be found among its subspaces.

We begin with the truth-value lattice, which has variously been called $\mathbf{2}$ or $\mathbb{T}$. In this chapter we call it $\mathbb{T}$ and think of it as having four elements $\{\bot, tt, ff, \top\}$ thus:



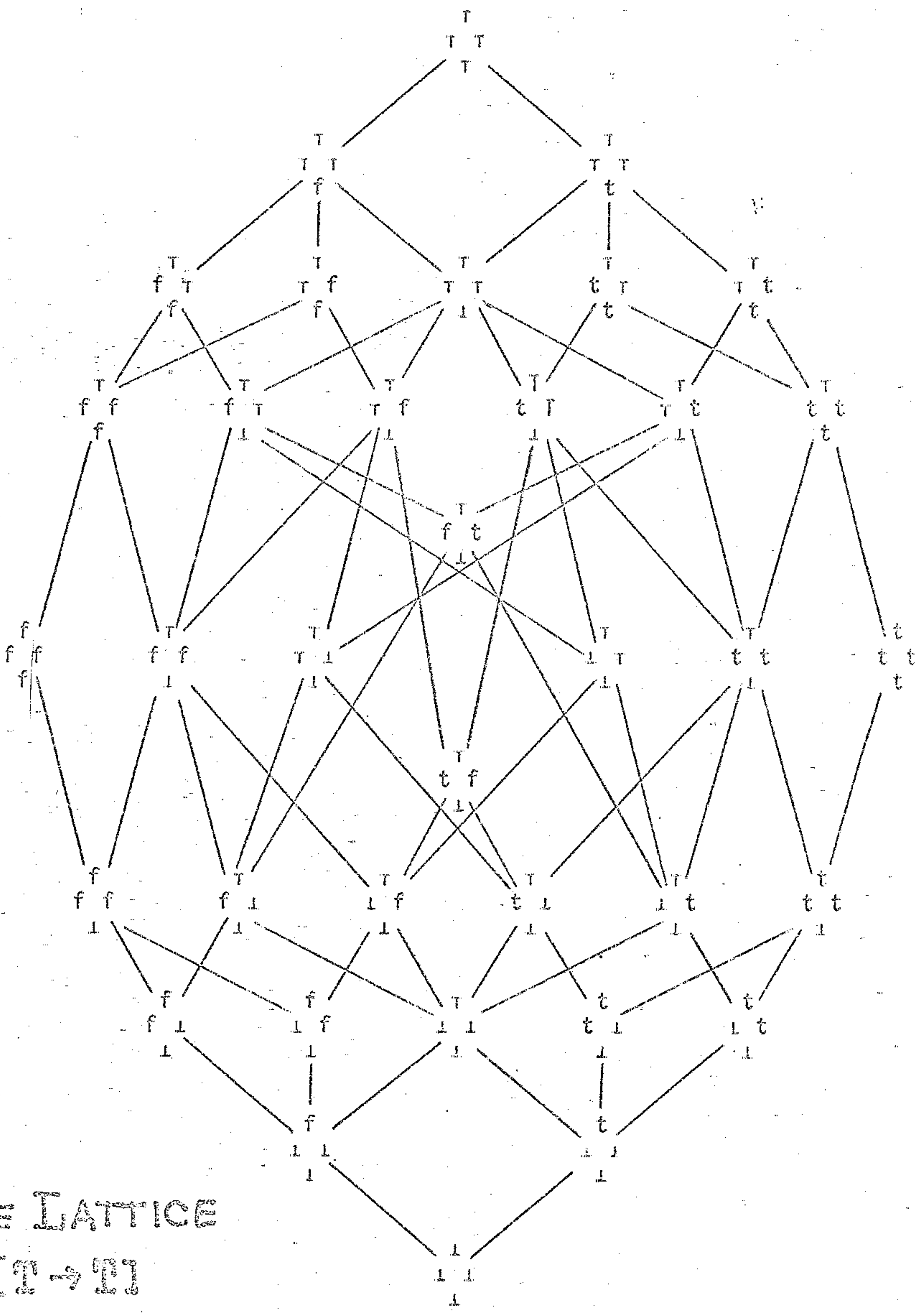
THE LATTICE $\mathbb{T}$

Here tt = true and ff = false. Of course $tt \not\sqsubseteq ff$ and $ff \not\sqsubseteq tt$ but $tt \sqcup ff = \top$ and $tt \sqcap ff = \bot$. (Remember that "truth" functions must always be monotonic.)

What we are going to do is to begin with $\mathbb{T}_0 = \mathbb{T}$ and iterate the function-space construction. Inductively:

$$\mathbb{T}_{n+1} = [\mathbb{T}_n \to \mathbb{T}_n].$$

This gives only a small selection of higher-type spaces over $\mathbb{T}$, but we shall show how all the others can be embedded in these. All the $\mathbb{T}_n$ are <u>finite</u> lattices (in which continuous = monotonic) but they grow in size very quickly. The lattice $\mathbb{T}_1$ is shown in the next figure.



THE LATTICE
$[\mathbb{T} \to \mathbb{T}]$

In the diagram on p. 59 the nodes represent functions (there are 36 in all). A (monotonic) function is notated by recording its values in a position corresponding to the argument. Each descending line represents a reduction of one value. It would be quite impossible to draw the figure for T_2; nevertheless we can think about these lattices and their relationships.

In a very crude way we can regard T_0 as being a part of T_1 by defining mappings:

$$T_0 \underset{j_0}{\overset{i_0}{\rightleftarrows}} T_1 \, ,$$

where

$$i_0(x) = \lambda y \in T_0 . \, x$$

$$j_0(f) = f(\perp) .$$

This is just the projection pair K, M of the exercise on p. 53. Next we move up the types by induction:

$$T_n \underset{j_n}{\overset{i_n}{\rightleftarrows}} T_{n+1} \, ,$$

where

$$i_{n+1} = \lambda f \in T_{n+1} \cdot i_n \circ f \circ j_n$$

$$j_{n+1} = \lambda F \in T_{n+2} \cdot j_n \circ F \circ i_n \,.$$

EXERCISE. Prove by induction that all the pairs i_n, j_n are projections.

In a way this definition is such that first crude approximation of T_0 into T_1 gets better and better as we go up to T_n and T_{n+1}. The way to think of the function j_n is that, for $f \in T_{n+1}$, the element $j_n(f) \in T_n$ is the best approximation to f that can be found in T_n. Of course the approximations are never perfect since T_{n+1} is a much larger space than T_n. In order to attain some kind of perfection, we must pass to the limit space.

DEFINITION. The infinite type space T_∞ is the space of all infinite sequences $\langle x_n \rangle_{n=0}^{\infty}$, where $x_n \in T_n$ and $x_n = j_n(x_{n+1})$ for all $n = 0, 1, 2, \ldots$ The partial ordering is given by:
$$\langle x_n \rangle_{n=0}^{\infty} \sqsubseteq \langle y_n \rangle_{n=0}^{\infty} \text{ iff } x_n \sqsubseteq y_n \text{ for all } n.$$

THEOREM. T_∞ is a continuous lattice.

This result is proved in the paper "Continuous Lattices". The calculations for showing that T_∞ is a complete lattice are not difficult - the continuous property, however, is more involved. The analysis of the construction shows that each T_n is a <u>projection</u> of T_∞ by functions

$$T_n \underset{j_{\infty n}}{\overset{i_{n\infty}}{\rightleftarrows}} T_\infty$$

where

$$i_{n\infty}(a) = \langle \ldots, j_{n-2}(j_{n-1}(a)), j_{n-1}(a), \overset{n^{\text{th}}\text{ place}}{\overset{\downarrow}{a}}, i_n(a), i_{n+1}(i_n(a)), \ldots \rangle$$

$$j_{\infty n}(\langle x_m \rangle_{m=0}^{\infty}) = x_n .$$

EXERCISE. Prove that $i_{n\infty}$, $j_{\infty n}$ form a projection and that:

$$i_{(n+1)\infty} \circ i_n = i_{n\infty} ,$$

$$j_n \circ j_{\infty(n+1)} = j_{\infty n} .$$

In view of these projections we shall now regard T_n <u>as</u> a subspace of T_{n+1} and all T_n <u>as</u> subspaces of T_∞: $T_0 \subseteq T_1 \subseteq \ldots \subseteq T_n \subseteq T_{n+1} \subseteq \ldots \subseteq T_\infty$.

The infinite-type space is more than the union of the finite type spaces: it is the completion of the union. The union of the finite spaces, however, is important

THEOREM. The union $\bigcup_{n=0}^{\infty} T_n$ of the elements of finite type consists exactly of the isolated elements of T_∞. Further, the isolated elements are dense in T_∞ in the sense that every element of T_∞ is the limit of elements of finite type.

The content of this theorem can be made clearer with the aid of some notation. In effect we are identifying T_n with its image $i_{n\infty}(T_n)$ in T_∞. If $x = \langle x_n \rangle_{n=0}^{\infty}$ is an element of T_∞, then by the identification x_n is identified with $i_{n\infty}(x_n)$. Hence we can regard x_n as the projection of x into the n^{th} subspace. That is, if $x \in T_\infty$, then we regard for simplicity

$$x_n \in T_n \subseteq T_\infty.$$

These type subscripts satisfy several useful laws. In particular $\lambda x. x_n$ is a continuous function.

THE LAWS OF PROJECTION. For $x \in T_\infty$:

(i) $x_n \in T_n$

(ii) $x_n \sqsubseteq x_{n+1}$

(iii) $x_{nm} = x_{\min(n,m)}$

(iv) $x = \bigsqcup_{n=0}^{\infty} x_n$

In formula (iv) the $\sqcup$ is taken in T_∞ and it shows why isolated elements are dense. As a consequence, if $e \in T_n$ and $e \sqsubseteq x$, then $e \sqsubseteq x_n$. Next we must consider application.

THE LAWS OF APPLICATION. For $x, y \in T_\infty$:

(i) $x_0 = x_0(y) = x(\perp)_0$

(ii) $x_{n+1} = \lambda y.\, x(y_n)_n$

(iii) $x(y) = \bigsqcup_{n=0}^{\infty} x_{n+1}(y_n)$

These laws require some explanation. Recall that $T_{n+1} = T_n \to T_n$. Thus the combination $x_{n+1}(y_n)$ makes sense and it belongs to T_n. Hence the limit makes sense also. Now application on $T_{n+1} \times T_n$ into

T_n is a continuous function; therefore $x_{n+1}(y_n)$ is a continuous function on $T_\infty \times T_\infty$. Thus whatever function is defined by the limit in (iii), it is a continuous function. We would like to call this function <u>application</u>. Now suppose $f : T_\infty \to T_\infty$ is an arbitrary continuous function. Then

$$f_{(n+1)} = \lambda y \in T_n . \ f(y)_n$$

makes sense and is in T_{n+1}. We can define $f_{(0)} = f(\bot)_0$. Then $f_{(n)} \in T_n$ and $f_{(n)} = j_n(f_{(n+1)})$, as can easily be checked. Thus

$$\hat{f} = \bigsqcup_{n=0}^{\infty} f_{(n)} \in T_\infty$$

and $\hat{f}_n = f_{(n)}$. Furthermore

$$\begin{aligned}
\hat{f}(y) &= \bigsqcup_{n=0}^{\infty} f_{(n+1)}(y_n) \\
&= \bigsqcup_{n=0}^{\infty} f(y_n)_n \\
&= \bigsqcup_{n=0}^{\infty} \bigsqcup_{m=0}^{\infty} f(y_n)_m \\
&= \bigsqcup_{n=0}^{\infty} f(y_n) = f\Big(\bigsqcup_{n=0}^{\infty} y_n\Big) = f(y).
\end{aligned}$$

This calculation shows that the application defined by (iii) (using $\hat{f}$) gives the same as ordinary application (using f). Thus we can safely identify f with $\hat{f}$. In outline we have proved:

THEOREM. $T_\infty = [T_\infty \to T_\infty]$.

Laws (i) and (ii) are recursion equations which help calculate the projections. For (i) we reason that the elements of T_0 are identified with the constant functions. That is:

$$\top(x) = \top$$

$$tt(x) = tt \qquad ff(x) = ff$$

$$\bot(x) = \bot$$

Thus for $x_0 \in T_0$, we have $x_0(y) = x_0$. Now $x_0 \sqsubseteq x$, so $x_0(\bot) \sqsubseteq x(\bot)$. But then $x_0 \sqsubseteq x(\bot)$ and $x_{00} \sqsubseteq x(\bot)_0$. This gives $x_0 \sqsubseteq x(\bot)_0$. The other way:

$$x(\bot)_0(y) = x(\bot)_0 \sqsubseteq x(\bot) \sqsubseteq x(y)$$

Thus $x(\bot)_0 \sqsubseteq x$, so $x(\bot)_{00} \sqsubseteq x_0$; whence $x(\bot)_0 \sqsubseteq x_0$.

Formula (ii) is the same as:

$$x_{n+1} = \lambda y \in T_n \,.\, x(y)_n ,$$

which means that the objects of type n+1 are regarded as mappings from type n into type n. This is just another form of the definition of j_{n+1}.

This discussion also shows why general λ-abstraction is possible in T_∞. If the expression (⋯ x ⋯ y ⋯) is continuous in all its variables, then in ordinary terms

$$f = \lambda y .\, (\cdots x \cdots y \cdots)$$

is a continuous function in $[T_\infty \to T_\infty]$ (and the expression is still continuous in x.) But we can say this means $\hat{f} \in T_\infty$, and the expression is still continuous in x. Thus by the same method:

$$\lambda x .\, \lambda y .\, (\cdots x \cdots y \cdots)$$

means something in T_∞. This notion of λ-abstraction is good, because

$$(\lambda x .\, \lambda y .\, (\cdots x \cdots y \cdots))(a)(b) =$$
$$(\cdots a \cdots b \cdots),$$

for all $a, b \in T_\infty$.

EXERCISE. Consider the combinator:

$$S = \lambda f.\lambda g.\lambda x.\ f(x)(g(x))$$

By using the laws of application show that

$$S_{n+3} = \lambda f.\lambda g.\lambda x.\ f_{n+2}(x_n)(g_{n+1}(x_n)).$$

We shall prove some theorems of interest for the λ-calculus using these laws.

THEOREM. (Scott) Let $\Delta = \lambda x.\ x(x)$. Then in T_∞ we have

$$\Delta(\Delta) = \perp.$$

Proof: We must calculate.

$$\Delta_0 = \Delta(\perp)_0 = \perp(\perp)_0 = \perp_0 = \perp$$

Thus $\Delta_0(\Delta_0) = \Delta_0$

Next $\Delta_{n+1}(\Delta_{n+1}) = \Delta(\Delta_{(n+1)n})_n = \Delta(\Delta_n)_n$

$$= \Delta_n(\Delta_n)_n = \Delta_n(\Delta_n)$$

because $\Delta_n(\Delta_n) \in T_n$. By induction we see

$$\Delta_n(\Delta_n) = \perp \text{ for } \underline{\text{all}}\ n.$$

But $\Delta(\Delta) = \bigsqcup_{n=0}^{\infty} \Delta_n(\Delta_n) = \bigsqcup_{n=0}^{\infty} \perp = \perp.$

THEOREM. (David Park). Let $F = \lambda x. f(x(x))$. Then in T_∞ we have

$$F(F) = Y(f).$$

("The λ-calculus fixed point is the same as the mathematical fixed point.")

Proof: First $F(F) = f(F(F))$, by conversion; hence $F(F)$ is a fixed point. The mathematical fixed point is the least one, so

$$Y(f) \sqsubseteq F(F).$$

Now calculate:

$$F_0 = F(\perp)_0 = f(\perp(\perp))_0 = f(\perp)_0 = f_0$$

So $\quad F_0(F_0) = f_0(f_0) = f_0 = f_0(\perp)$

Next $\quad F_{n+1}(F_{n+1}) = F(F_n)_n = f(F_n(F_n))_n$

$$= f_{n+1}(F_n(F_n))$$

because $F_n(F_n) \in T_n$. Therefore

$$F(F) = \bigsqcup_{n=0}^{\infty} F_n(F_n)$$

$$= \bigsqcup_{n=0}^{\infty} f_n(f_{n-1}(\cdots(f_1(f_0))\cdots))$$

$$\sqsubseteq \bigsqcup_{n=0}^{\infty} f^{n+1}(\perp) = Y(f).$$

THEOREM. (Christopher Wadsworth) Let $\mathcal{J} = Y(\lambda f . \lambda x . \lambda y . x(f(y)))$. Then $\mathcal{J}$ has no normal form by the ordinary rules of λ-conversion, BUT in D_∞ we have $\mathcal{J} = \mathcal{I}$.

Proof: Clearly:

$$\mathcal{J} = \lambda x . \lambda y . x(\mathcal{J}(y))$$

So

$$\mathcal{J} \sqsubseteq \mathcal{I} .$$

Now calculate (to prove inductively $\mathcal{J}(x_n) \sqsupseteq x_n$):

$$\mathcal{J}(x_0) = \lambda y . x_0(\mathcal{J}(y))$$

$$= \lambda y . x_0 = x_0$$

Next

$$\mathcal{J}(x_{n+1}) = \lambda y . x_{n+1}(\mathcal{J}(y))$$

$$\sqsupseteq \lambda y . x_{n+1}(\mathcal{J}(y_n))$$

$$\sqsupseteq \lambda y . x_{n+1}(y_n)$$

$$= \lambda y . x_{n+1}(y)$$

$$= x_{n+1} .$$

Therefore

$$\mathcal{J}(x) = \bigsqcup_{n=0}^{\infty} \mathcal{J}(x_n)$$

$$\sqsupseteq \bigsqcup_{n=0}^{\infty} x_n = x$$

So

$$\mathcal{J} \sqsupseteq \mathcal{I} .$$

PROPOSAL. What Church called λ-calculus based on α-β-η-conversion, I would now like to call λ-algebra. Not everything can be proved by algebra alone. These last three theorems show that more subtile equations require limiting arguments. It is the latter study that I would call λ-calculus—when one takes advantage of the infinite series representations that are available in T_∞.

Many other constructions can be done in T_∞ aside from those of the pure theory of λ-abstraction.

EXERCISE. Prove that $x_0 = tt$ iff $tt \sqsubseteq x$ and $x \neq \top$. Similarly for ff.

DEFINITION. (The Conditional)

$$(z \supset x, y) = \begin{cases} x \sqcup y & \text{if } z = \top \\ x & \text{if } tt \sqsubseteq z \neq \top \\ y & \text{if } ff \sqsubseteq z \neq \top \\ \bot & \text{otherwise.} \end{cases}$$

EXERCISE. Show that ⊃ is continuous.

EXERCISE. Investigate the "strict" conditional:

$$(z :\supset x, y) = (z \supset (z \supset x, \top), (z \supset \top, y))$$

EXERCISE. Show that

$$z_0 = (z \supset tt, ff).$$

DEFINITIONS.

$$(x, y) = \lambda z.\ (z \supset x, y)$$

$$fst = \lambda u.\ u(tt)$$

$$snd = \lambda u.\ u(ff)$$

EXERCISE. Prove that

$$fst((x,y)) = x,$$

$$snd((x,y)) = y,$$ and that the function $\lambda u.\ (fst(u), snd(u))$ is a projection.

The function $pair = \lambda x.\ \lambda y.\ (x, y)$ gives us ordered pairs within T_∞ in a very satisfactory way. In particular many constructions with cartesian products can now be done with retracts all within T_∞.

DEFINITIONS.

$$a \times b = \lambda u.\ (a(\mathrm{fst}(u)),\ b(\mathrm{snd}(u)))$$

$$a + b = \lambda u.\ (\mathrm{fst}(u) :\supset (tt,\ a(\mathrm{snd}(u))),\ (ff,\ b(\mathrm{snd}(u))))$$

$$a \to b = \lambda f.\ b \circ f \circ a$$

DEFINITION. If a is a retract, let us write $x \in a$ to mean that x is in the range of a. This is equivalent to $a(x) = x$, because the range of a retract is the same as the set of fixed points

EXERCISE. Prove that if a and b are retracts, then so is $a \times b$ and

$u \in a \times b$ iff $u = (x, y)$ where $x \in a$ and $y \in b$ for some x, y.

EXERCISE. Prove that $a + b$ is a retract if a and b are, and that

$u \in a + b$ iff $u = \perp$ or $u = \top$ or
$u = (tt, x)$ some $x \in a$ or
$u = (ff, y)$ some $y \in b$.

EXERCISE. Prove that $a \to b$ is a retract if a and b are, and that

$$u \in (a \to b) \text{ iff } u \circ a = u \text{ and for all } x \in a \text{ we have } u(x) \in b.$$

The above results make it possible to regard $a \times b$, $a + b$, $a \to b$ as subspace constructions in an intuitive way.

EXAMPLE. Let us write

$$\mathbb{T} = (\mathrm{tt}, \mathrm{ff})$$

Then $\mathbb{T} = \lambda z. z_0$ = the projection onto T_0. Now $\mathbb{T} \in T_\infty$ and we can use it as a constant in equations. In fact there is no reason not to solve such fixed-point equations as:

$$\mathbb{T}^* = Y(\lambda t. (\mathbb{T} \times t))$$

Thus

$$\mathbb{T}^* = \mathbb{T} \times \mathbb{T}^*.$$

One should investigate what subspace is represented by the retraction (projection?) $\mathbb{T}^*$.

CHAPTER VII. Semantics.

Each space has an appropriate language in which the structural properties of its elements can be described. Sometimes the language developed before the space was known — the λ-calculus is an example. Sometimes the language and its structured space of denotations will develop together. The point of the present study is to show the very large variety of spaces that can be constructed and to illustrate how the corresponding "calculus" can be understood. The usefulness for programming language semantics has been described in several publications ("The lattice of flow diagrams", "Lattice theory, data types, and semantics", "Mathematical concepts in programming language semantics", (with Strachey) "Toward a mathematical semantics for computer languages", (by Strachey) "Varieties of Programming language.") We shall not here have time to review any of the examples of those papers. Instead we shall look at a few cases closely connected with the more specific material of these notes

The first main point is that the infinite type space T_∞ may be regarded as a kind of universal space inside of which our constructions are carried out. Now, to make this suggestion practical, we must invent a language appropriate to T_∞. Actually, we have already done so in an informal way in the last chapter. We may now formalize the language and will find that it is something of a combination of the λ-calculus of Church and something of the language LISP of McCarthy. It is given as a mathematical language rather than as a programming language. However, many algorithms can be expressed in it.

THE LANGUAGE LAMBDA

$$\xi ::= \text{(the usual identifiers)}$$

$$\varepsilon ::= \xi \mid \bot \mid \top \mid \mathbf{tt} \mid \mathbf{ff} \mid \supset \mid \lambda\xi.\varepsilon \mid \varepsilon_0\varepsilon_1 \mid (\varepsilon)$$

(This grammar has been left ambiguous in order not to require endless parentheses. The last clause for expressions allows for parentheses to be used wherever necessary.)
If we let Id be the set of _identifiers_ and Exp the set of _expressions_, then the formal semantics can be given by means of a system of semantical equations about an evaluation function $\mathcal{E}$ that maps expressions and environments to denotations. Here the denotations (values) belong simply to T_∞ and the environments are mappings $\rho : \mathrm{Id} \to T_\infty$. We have:

THE SEMANTICAL EQUATIONS.

$$\mathcal{E} : Exp \to [[Id \to T_\infty] \to T_\infty]$$

$$\mathcal{E}(\xi)(\rho) = \rho(\xi)$$
$$\mathcal{E}(\perp)(\rho) = \perp$$
$$\mathcal{E}(\top)(\rho) = \top$$
$$\mathcal{E}(\mathbf{tt})(\rho) = tt$$
$$\mathcal{E}(\mathbf{ff})(\rho) = ff$$
$$\mathcal{E}(\supset)(\rho) = \lambda z.\lambda x.\lambda y.(z \supset x, y)$$
$$\mathcal{E}(\lambda\xi.\varepsilon)(\rho) = \lambda x.\ \mathcal{E}(\varepsilon)(\rho[x/\xi])$$
$$\mathcal{E}(\varepsilon_0\varepsilon_1)(\rho) = \mathcal{E}(\varepsilon_0)(\rho)(\mathcal{E}(\varepsilon_1)(\rho))$$
$$\mathcal{E}((\varepsilon))(\rho) = \mathcal{E}(\varepsilon)(\rho)$$

On the left side we have the symbols and on the right the denotations. The ρ is needed to handle assignments to variablas. The operation $\rho[x/\xi]$ means to change ρ in that x is assigned to ξ. By means of these equations, every expression has a value in T_∞ once the environment of assignments to the (free) variables is known. We can then define:

THE EQUIVALENCE RELATION

$$\mathcal{E}_0 \equiv \mathcal{E}_1 \quad \text{iff for all } \rho : Id \to T_0$$
$$\mathcal{E}(\mathcal{E}_0)(\rho) = \mathcal{E}(\mathcal{E}_1)(\rho)$$

In the last chapter we showed (informally) that many expressions are equivalent. For example, $\Delta(\Delta) \equiv \perp$. This result has been generalized by Christopher Wadsworth to give a characterization of all pure λ-expressions (those without $\top, t, f, \supset$) which are $\equiv \perp$.

In general a complete characterization of $\equiv$ is impossible because it is not a recursively enumerable relation between expressions. However, a great deal can be done *via* formal proofs as is shown by the work of de Bakker (on recursive procedures) and the recent work of Robin Milner. The following formalism has grown out of correspondence with Milner based on earlier (unpublished) suggestions of the author.

By a *formula* we understand either:

$$\mathcal{E}_0 \equiv \mathcal{E}_1 \quad \text{or} \quad \mathcal{E}_0 = \mathcal{E}_1$$

By a *sequent* we understand

$$\Gamma \vdash \Delta$$

where Γ and Δ are finite sets of formulas. A formula, say $\mathcal{E}_0 \equiv \mathcal{E}_1$, is *true* in an environment ρ iff

$$\mathcal{E}(\mathcal{E}_0)(\rho) \equiv \mathcal{E}(\mathcal{E}_1)(\rho)$$

(Similarly for $=$). A sequent is *valid* iff for all environments ρ, *if* all formulas in Γ are true in ρ, *then* at least one in Δ is true in ρ also. In this terminology $\mathcal{E}_0 \equiv \mathcal{E}_1$ means that $\vdash \mathcal{E}_0 \equiv \mathcal{E}_1$ is valid.

The idea of sequents is that of the well-known Gentzen calculus. The problem is to axiomatize as much of the valid sequents as possible. To a large extent the following axioms and rules are effective, where we write the sets of formulas without any $\{ , \}$.

THE AXIOMS AND RULES

(I) The Gentzen Rules with Equality and Substitution for Free Variables

(II) Partial Ordering:

$$\mathfrak{tt} \sqsubseteq \bot \vdash$$
$$\mathfrak{ff} \sqsubseteq \bot \vdash$$
$$\vdash \bot \sqsubseteq x$$
$$\vdash x \sqsubseteq \top$$
$$\vdash x \sqsubseteq x$$
$$x \sqsubseteq y,\ y \sqsubseteq x \vdash x = y$$
$$x \sqsubseteq y,\ y \sqsubseteq z \vdash x \sqsubseteq z$$
$$\mathfrak{tt} \sqsubseteq z,\ \mathfrak{ff} \sqsubseteq z \vdash \top \sqsubseteq z$$
$$z \sqsubseteq \mathfrak{tt},\ z \sqsubseteq \mathfrak{ff} \vdash z \sqsubseteq \bot$$

(III) The Conditional Expression:

$$\mathfrak{tt} \sqsubseteq z \vdash \supset(z)(x)(y) = x\ ,\ z = \top$$
$$\mathfrak{ff} \sqsubseteq z \vdash \supset(z)(x)(y) = y\ ,\ z = \top$$
$$\vdash \supset(z)(x)(y) = \bot\ ,\ \mathfrak{tt} \sqsubseteq z,\ \mathfrak{ff} \sqsubseteq z$$
$$\vdash \supset(\top)(x)(x) = x$$

(IV) Application and Abstraction:

$\vdash \mathbb{tt}(x) = \mathbb{tt}$

$\vdash \mathbb{ff}(x) = \mathbb{ff}$

$\vdash (\lambda x.\varepsilon)(x) = \varepsilon$

$x \equiv y \vdash f(x) \equiv f(y)$

$$\frac{\Delta \vdash f(x) \equiv g(x),\ \Gamma}{\Delta \vdash f \equiv g,\ \Gamma}$$ where x is not free in Δ, Γ

(V) Induction

$$\frac{\begin{array}{l}\Delta \vdash f(\bot) = g(\bot),\ \Gamma \\ \Delta,\ f(x) = g(x) \vdash f(F(x)) = g(F(x)),\ \Gamma\end{array}}{\Delta \vdash f(Y(F)) = g(Y(F)),\ \Gamma}$$

where x is not free in Δ, Γ.

$\mathbb{T} \equiv x,\ (x \rightarrow x) \equiv x \vdash I \equiv \varepsilon$.

where by definition:

$Y = \lambda F.(\lambda x.F(x(x)))(\lambda x.F(x(x)))$

$\mathbb{T} = \lambda x.\supset(x)(\mathbb{tt})(\mathbb{ff})$

$\rightarrow\ = \lambda a.\lambda b.\lambda f.\lambda x.\ b(f(a(x)))$

$I = \lambda x.x$

There is not time here for a discussion of what is formally provable from these axioms and rules, but they were stated so as to show that such axioms need not be too complicated. The axioms are also geared exactly to the construction of T_∞. The only place we have taken advantage of the special character of T_∞ is in the statement of the induction rule: here we used the λ-definition of the Y-operator. We shall discuss the question of formal definability

DEFINITION. An element $d \in T_\infty$ is said to be <u>definable</u> if there is an expression $\mathcal{E}$ of LAMBDA without free variables such that

$$d = \mathcal{E}(\varepsilon)(\rho)$$

for some (= all) environments ρ.

This leads us to the statement of the computability thesis:

THESIS. The <u>computable</u> elements of T_∞ are exactly the LAMBDA-definable elements.

This is the analogue for T_∞ of Church's Thesis. Either one uses a numerical equation calculus for partial recursive functions or one speaks about λ-definability of partial recursive functions. Here we have extended the idea to what is defined by arbitrary (closed) expressions. To appreciate the reasonableness of this identification of computability, one should remark first, as shown in the last chapter, that all finite (= isolated) elements are LAMBDA-definable in T_∞. Next if d is any definable element, we can rewrite its definition as

$$d = \bigsqcup_{n=0}^{\infty} \varepsilon^{(n)}$$

where, informally speaking, we have "subscripted" all subexpressions of its definition by the projection to $\mathbb{T}_n$. The value of $\mathcal{E}^{(n)}$ will then be an effectively computable element of the *finite* space $\mathbb{T}_n$. Thus d is represented in $\mathbb{T}_\infty$ as an *effective limit*.

Finally (by an appeal to Church's Thesis) if d is any element represented by an effective limit:

$$d = \bigsqcup_{n=0}^{\infty} e^{(n)},$$

where $e^{(n)} \in \mathbb{T}_n$, then we can define d by a LAMBDA expression. Why? Because we can Gödel number all the isolated elements and can then use a numerical function to choose effectively the proper $e^{(n)}$. More precisely, we must represent the integers in LAMBDA and then find an expression $\mathcal{E}$ such that

$e^{(n)}$ is defined by $\varepsilon(\bar{n})$ where $\bar{n}$ is the expression representing n. We sketch how to do this.

DEFINITION. We represent integers by this scheme:

$$\bar{0} = (\#, \bot)$$

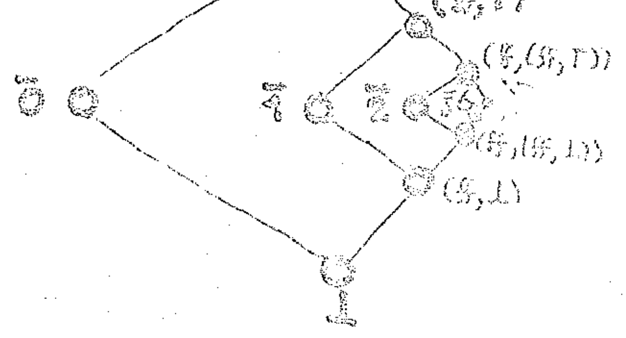
$$\overline{n+1} = (\#, \bar{n})$$

There is a retraction for the integers we can get in two stages:

DEFINITION.

$$\hat{N} = \bot + \hat{N}$$

This is short for a fixed-point definition. The retract looks like this:



This does not give quite the right space because there there the extraneous elements

$$(ff, (ff, (ff, \cdots (ff, \perp) \cdots)))$$

and the infinite limit point:

$$\vec{ff} = (ff, \vec{ff}).$$

These are eliminated by a test-function:

DEFINITIONS.

$$\mathit{test} = \lambda n.(\mathit{fst}(n) \supset tt, \mathit{test}(\mathit{snd}(n)))$$

$$\mathbb{N} = \lambda n.(\mathit{test}(n) \supset n, \perp)$$

$$\mathit{suc} = \lambda n.\mathbb{N}((ff, \mathbb{N}(n)))$$

$$\mathit{iszero} = \lambda n.\mathit{fst}(\mathbb{N}(n))$$

$$\mathit{pred} = \lambda n.\mathit{snd}(\mathbb{N}(n))$$

With these basic definitions one can now proceed along standard lines to show that all partial recursive functions of integers are LAMBDA definable. Finally it only remains to show how to take the <u>limit</u>.

DEFINITION.

$$\bigsqcup = \lambda f . (f(\bar{0}) \sqcup \bigsqcup (f \circ \mathrm{suc}))$$

This fixed-point definition gives us

$$\bigsqcup(f) = \bigsqcup_{n=0}^{\infty} f(\bar{n}),$$

and thus makes it possible to define a wide variety of other elements.

Computability on other structures can now be discussed by using retracts (definable retracts) and the restrictions of definable functions to these subspaces. Thus the whole theory is considerably unified.

ADDENDUM. By an oversight the formulas for the isolated elements had not been included in Chapter VI. The idea is based on the "step function" mentioned on p. 45. In case e is an isolated element we have

$$\vec{e}(e,e')(x) = \begin{cases} e' & \text{if } e \sqsubseteq x \\ \perp & \text{otherwise} \end{cases}$$

We note that the special case

$$[e] = \vec{e}(e, \top)$$

is essentially all that is needed because

$$\vec{e}(e,e') = [e] \sqcap \lambda x. e'$$

Now if $e, e' \in \mathbb{T}_n$, it is easy to see that $\vec{e}(e,e') \in \mathbb{T}_{n+1}$. Further, if $f \in \mathbb{T}_{n+1}$, then

$$f = \bigsqcup \{\vec{e}(e,e') : e' \sqsubseteq f(e), e, e' \in \mathbb{T}_n\}$$

which is a <u>finite union</u> because $\mathbb{T}_n$ is finite. Indeed. $f = \bigsqcup \{\vec{e}(e, f(e)) : e \in \mathbb{T}_n\}$

GENERATING THE BASIS. The isolated elements of T_∞ are thus generated from $\perp$, tt, ff by iteration of the operations $e \sqcup e'$ and $\vec{e}(e,e')$.

It should be noted that as an operat[ion] $\vec{e}(e,e')$ is not monotonic in e, even though the result is a perfectly goo[d] continuous function.

THE RECURSION EQUATIONS. We may now calculate $[e]$ of any isolate[d] element e by looking at how it is generated and using these equations:

$$[\perp] = \lambda x.\top$$

$$[\top] = \lambda x.\, x \supset \perp, \perp$$

$$[tt] = \lambda x.\, x \supset \top, \perp$$

$$[ff] = \lambda x.\, x \supset \perp, \top$$

$$[e \sqcup e'] = [e] \sqcap [e']$$

$$[\vec{e}(e,e')] = \lambda f\; [e'](f(e))$$

As a consequence we see why each such element is LAMBDA-definabl[e]