From the introduction to

The Combinatory Programme

by

E. Engeler

in collaboration with
K. Aberer, O. Gloor, M. von Mohrenschildt,
D. Otth, G. Schwärzler, T. Weibel

published by Birkhäuser, Boston (etc.), 1995, 142 pp.
(Progress in theoretical computer science)
ISBN 0-8176-3801-6.

In the fall of 1928 a young American turned up at the Mathematical Institute of Göttingen, a mecca of mathematicians at the time; he was a young man with a dream and his name was H. B. Curry. He felt that he had the tools in hand with which to solve the problem of foundations of mathematics once and for all: his was an approach that came to be called "formalist", and embodied a technique that later became known as Combinatory Logic. Closest to his ideas was the work of Schönfinkel on the "building blocks of mathematical logic" [Schönfinkel], and the man who best knew about this was Bernays, the main collaborator of Hilbert at Göttingen on the latter's foundational programme. This is why Curry went there to submit his thesis [Curry].

The present book is, however, not a history of the actual, very rich, development of Curry's ideas; it is a report on a mathematical programme pursued at the ETH Zurich which was, in part, inspired by some of these ideas, but greatly influenced by later developments, particularly by universal algebra, logic programming, computer algebra, and numerical analysis, and therefore embedded in contemporary concerns. The relation of combinatory logic to these areas determine the structure of this book, in particular its technical chapters.

1 From protologics to combinatory algebras

Richard Dedekind was probably the first mathematician to try to base the development of mathematics on a theory of pure thought. In his delightful and influential booklet "Was sind und was sollen die Zahlen?" (1888) he motivates his approach quite generally by pronouncing – somewhat sententiously – that in science nothing should be accepted without proof. In particular, how should one prove that there are infinite totalities such as the set of natural numbers? Philosophical logic – at the time a somewhat dried-up subject – and mathematical logic – then in its infancy – would have been the "theory of pure thought" on which the basis had to be laid. Dedekind did not avail himself of them but found nevertheless a somewhat plausible argument for the existence of classes which can be mapped one-to-one on a proper subclass, i.e. infinite classes. His argument [Dedekind] (theorem 66) proposes as this class "the totality of all things which may be object of my thought". This class S is mapped into itself by associating to each object of thought x an object $d(x)$, namely the thought that x is an object of thought; obviously the thought about the thought is different from the thought itself, so d is indeed monomorphic. The self of the thinking subject is clearly an object of thought and not itself of the form $d(x)$. Thus S is mapped one-to-one onto a proper subclass.

Dedekind's argument was not accepted at the time as a technical proof (as is documented in the re-edition of his booklet in the collected works), nor would it stand as one today. But the idea of creating a theory about thoughts, objects of thoughts and how they apply to each other is a tempting one. In fact, if logic is to be the science of correctly dealing with thought-objects, such a theory would in some sense have to be a part, if not a preliminary, to logic: a protologic.

How could one, today, obtain such a protological theory? In the spirit of contemporary mathematics we need to talk about a structure, i.e. a set of elements and some operations on them. The "nature" of the elements is not the primary concern, although it might be helpful to have a concrete example. Elements exist and have certain properties in the theory exactly in so far as these follow from some basic assumptions about the structure, the axioms.

Returning to objects of thought and how to apply them to each other, consider the thought $\ulcorner$wisdom$\urcorner$ and the thought-object $\ulcorner$Sokrates$\urcorner$. Applying one to the other, we obtain the thought $\ulcorner$the wisdom of Sokrates$\urcorner$. Formally,

$$\ulcorner\text{wisdom}\urcorner \cdot \ulcorner\text{Sokrates}\urcorner = \ulcorner\text{the wisdom of Sokrates}\urcorner.$$

Among the everyday objects of thought we find not only individuals and their properties but also relationships between objects, e.g., the thought that there is an analogy between computers and the brain. First-order logic would express this by

$$\text{analog(computer,brain)},$$

i.e. with a binary predicate. The thought in question is really a composite thought, at least if we consider the notion of application of one thought(-object) to another as fundamental. It is the thought $\ulcorner$to be analogous to a computer$\urcorner$ applied to $\ulcorner$brain$\urcorner$ where the first thought-object is itself the result of applying the concept $\ulcorner$analogy$\urcorner$ to $\ulcorner$computer$\urcorner$. Thus, altogether, our example is

$$(\ulcorner\text{analogy}\urcorner \cdot \ulcorner\text{ computer}\urcorner) \cdot \ulcorner\text{ brain}\urcorner.$$

This thought may itself be applied to the idea of the simulation on the computer of visual inputs to the brain, similarly expressed by

$$(((\ulcorner\text{simul}\urcorner \ \cdot \ \ulcorner\text{computer}\urcorner) \cdot \ \ulcorner\text{input}\urcorner) \cdot \ \ulcorner\text{brain}\urcorner)$$

yielding altogether the thought "to consider the computer-simulation of visual input into the brain as an application of the analogy between the computer and the brain", in shorthand

$$((a \cdot c) \cdot b) \cdot (((s \cdot c) \cdot i) \cdot b).$$

This particular form of combination of the five thought-objects a, b, c, i, s is one that may be found in very many contexts, one example being "the idea of reducing (r) chemistry (c) to physics (p), as applied to using quantum theory (q) in chemistry to explain a mechanism of bonding (b) by physics":

$$((r \cdot c) \cdot p) \cdot (((q \cdot c) \cdot b) \cdot p).$$

The thought of combining five thought-objects in just this form is itself without question a legitimate object of thought, let us denote it by "$\boldsymbol{V}$". Its defining property is

$$(((((\boldsymbol{V} \cdot x_1) \cdot x_2) \cdot x_3) \cdot x_4) \cdot x_5) = ((x_1 \cdot x_2) \cdot x_3) \cdot (((x_4 \cdot x_2) \cdot x_5) \cdot x_3),$$

which we write, conveniently suppressing parentheses to the left,

$$\boldsymbol{V} \cdot x_1 \cdot x_2 \cdot x_3 \cdot x_4 \cdot x_5 = (x_1 \cdot x_2 \cdot x_3) \cdot (x_4 \cdot x_2 \cdot x_5 \cdot x_3).$$

Thus we have made the thought formulated as an applicative expression on the right hand side, into a though-object which, whatever other thought-objects we may wish to consider later, would have to be present among the elements of the protological universe. The principle by which we introduced $\boldsymbol{V}$ is the

Principle of combinatory abstraction

For every applicative expression $t(x_1, \ldots, x_n)$ there exist $\boldsymbol{T}$ for which we have the equation

$$\boldsymbol{T} \cdot x_1 \cdot x_2 \cdots x_n = t(x_1, \ldots, x_n)$$

It expresses the fact that every pattern t of combining applications of thoughts to each other is itself an object $\boldsymbol{T}$ of thought.

Abstracting for a moment from the context through which we were led to the above principle, and returning to a contemporary algebraic point of view: what we have here is simply an axiom-scheme which specifies the basic property of a type of algebraic structure; just as the axioms of group theory specify the class of groups. Algebraic structures with one binary operation – application – satisfying the principle of combinatory abstraction are called *combinatory algebras*.

Are there in fact combinatory algebras? Experience with unfettered principles of abstraction – see Frege's comprehension principle and Russell's paradox – teaches us that it might be difficult to establish the existence of (non-trivial) combinatory algebras. If, following Hilbert, existence is equated with consistency, a positive answer is given by following the spirit of his program: Define combinatory logic by the axioms of equality together with the axiom scheme of combinatory abstraction and prove this logical system formally consistent. The proof [Church and Rosser] employs no stronger tools than primitive recursive arithmetic.

In contrast with this logical proof theoretic approach, the algebraic approach is set-theoretic in its basic concepts: to obtain a combinatory algebra, one needs to specify a set (in some set-theoretic universe) and explicitly define the operation of application for any two of its elements. Our favorite construction is that of the graph model [Engeler81], (see below).

The approach taken in this book is a programmatic mixture of the axiomatic and the set-theoretic. For experience and confidence with the proposed constructs we often rely on set-theoretic examples, and then try to abstract away this scaffolding in order to retain a purely axiomatic edifice.

Every combinatory algebra, by whichever road we come to consider it, has those elements whose existence is postulated by the principle of combinatory abstraction. Following Curry, these elements are called combinators and conventionally denoted by boldface capitals. Here are some examples:

$$\boldsymbol{K} \cdot x \cdot y = x,$$

$$\boldsymbol{D} \cdot x \cdot y = x \cdot (y \cdot y),$$

$$\boldsymbol{Y} \cdot x = (\boldsymbol{D} \cdot x) \cdot (\boldsymbol{D} \cdot x),$$

$$\boldsymbol{S} \cdot x \cdot y \cdot z = (x \cdot z) \cdot (y \cdot z),$$

$$\boldsymbol{I} \cdot x = x,$$

$$\boldsymbol{B} \cdot x \cdot y \cdot z = x \cdot (y \cdot z).$$

The combinator $\boldsymbol{Y}$ has been called "paradoxical" because, like Russell's paradox, it can be used to dash utopian hopes of letting protologic be a foundation of logic and mathematics. If indeed protologic – as embodied in combinatory algebra – admitted the usual logical connectives as elements, as thought-objects as it were, there would be an element N which stands for the thought $\ulcorner$the negative$\urcorner$. Thus N, applied to the concept $\ulcorner$true$\urcorner$ (if there were one) would yield $\ulcorner$false$\urcorner$; for $\ulcorner$good$\urcorner$ it would yield $N \cdot \ulcorner$ good$\urcorner = \ulcorner$ evil$\urcorner$, etc. Now, consider $\boldsymbol{Y} \cdot N$ and evaluate

$$\boldsymbol{Y} \cdot N = \boldsymbol{D} \cdot N \cdot (\boldsymbol{D} \cdot N) = N \cdot ((\boldsymbol{D} \cdot N) \cdot (\boldsymbol{D} \cdot N)) = N \cdot (\boldsymbol{Y} \cdot N).$$

Therefore the thought-object $\boldsymbol{Y} \cdot N$ would be equal to its own negation!

The positive aspect of $\boldsymbol{Y}$ is captured by its usual name "fixpoint combinator": Exactly as above for N, the application of $\boldsymbol{Y}$ to any element f provides for a fixpoint of φ_f when φ_f is considered to be a mapping obtained by left multiplication with f:

$$\begin{aligned} \varphi_f(x) &= f \cdot x, \\ \varphi_f(\boldsymbol{Y} \cdot f) &= \boldsymbol{Y} \cdot f \end{aligned}$$

It is a very powerful concept, one that in a sense has inspired many mathematical developments, such as recursion theory, indeed also some literary productions, such as Hofstadter's popular book.

Another illustration of the limitations of protologics, and therefore of any attempt to build upon "pure" logic is provided by the following generalization of Rice's recursion theoretic theorem [Rice]. It deals with attempts to single out a set of elements in a combinatory algebra by means of an either/or operator. Such an operator E would have two possible values if applied to an element x, either $\boldsymbol{K}$ or $\boldsymbol{K} \cdot \boldsymbol{I}$. The reason for taking $\boldsymbol{K}$ and $\boldsymbol{K} \cdot \boldsymbol{I}$ is clear: The value of $E \cdot x$ can be used to make a choice between any two objects, say u and v, namely, if E decides that x is of the first kind then

$$(E \cdot x) \cdot u \cdot v = \boldsymbol{K} \cdot u \cdot v = u,$$

if it is of the second kind, then

$$(E \cdot x) \cdot u \cdot v = \boldsymbol{K} \cdot \boldsymbol{I} \cdot u \cdot v = \boldsymbol{I} \cdot v = v.$$

Assume now that there are elements a and b of the first and second kind respectively, and define, using combinatory abstraction,

$$M \cdot x = (E \cdot x) \cdot b \cdot a.$$

Let c be a fixpoint of M, for example $c = \boldsymbol{Y} \cdot M$. Then E can neither decide that c is of the first kind, for then $c = M \cdot c = (E \cdot c) \cdot b \cdot a = \boldsymbol{K} \cdot b \cdot a = b$, nor can it decide that it is of the second kind, for then $c = M \cdot c = (E \cdot c) \cdot b \cdot a = \boldsymbol{K} \cdot \boldsymbol{I} \cdot b \cdot a = \boldsymbol{I} \cdot a = a$, while a and b were, as we recall, of the second, respectively of the first, kind.

Therefore,

Lemma 1.1 *A combinatory decision operator E can only divide the combinatory algebra to which it belongs into the empty set and itself.*

For creating mathematical structures strictly inside combinatory algebras we would therefore have to look for a different kind of mechanism.

2 A brief recapitulation of combinatory algebra

The principle of combinatory abstraction provides for a infinitude of combinators, but in fact two suffice to compose all others.

[Schönfinkel]
Lemma 2.1 *For every combinator $\boldsymbol{T}$ there exists an applicative expression $t(\boldsymbol{S},\boldsymbol{K})$ in $\boldsymbol{S}$ and $\boldsymbol{K}$ alone such that $\boldsymbol{T}=t(\boldsymbol{S},\boldsymbol{K})$.*

The proof, by induction on the structure of the formula which defines $\boldsymbol{T}$ by combinatory abstraction, is straightforward; the need for exactly these two combinators $\boldsymbol{S}$ and $\boldsymbol{K}$ develops in the proof.

Combinatory algebras have a surprising wealth of elements and operations. Most of these were originally discovered, mainly by Church [Church] and Kleene [Kleene] hand in hand with the development of recursion theory. The operation of pairing and its inverses are defined by

$$\begin{array}{lcll} \boldsymbol{P}\cdot u\cdot v\cdot x & = & x\cdot u\cdot v: & \boldsymbol{P}\cdot u\cdot v \text{ is the pair } \langle u,v\rangle, \\ \boldsymbol{P}_1 u & = & u\cdot \boldsymbol{K}: & \boldsymbol{P}_1 \text{ its first component,} \\ \boldsymbol{P}_2 u & = & u\cdot(\boldsymbol{K}\cdot\boldsymbol{I}): & \boldsymbol{P}_2 \text{ its second component;} \end{array}$$

and we verify for example

$$\boldsymbol{P}_2(\boldsymbol{P}uv)=(\boldsymbol{P}uv)(\boldsymbol{K}\cdot\boldsymbol{I})=\boldsymbol{K}\cdot\boldsymbol{I}\cdot u\cdot v=v.$$

Pairing can be used, just as in axiomatic set theory, to define each one of the natural numbers by a combinator, its combinatory numeral. To wit:

$$\boldsymbol{0}=\boldsymbol{I},\boldsymbol{1}=\boldsymbol{P}\cdot\boldsymbol{0}\cdot\boldsymbol{K},\ldots,\boldsymbol{5}=\boldsymbol{P}\cdot\boldsymbol{4}\cdot\boldsymbol{K},\ldots$$

The successor of a number is uniformly given by the next-number combinator

$$\boldsymbol{N}\cdot x=\boldsymbol{P}\cdot x\cdot\boldsymbol{K}.$$

This is a simple example of an arithmetic function representable by a combinator. The general fact is this:

Theorem 2.2 *For every partial recursive function $f:\mathbb{N}^n\to\mathbb{N}$ there exists a combinator $\boldsymbol{F}$ which numeralwise represents it, i.e. $\boldsymbol{F}\cdot\,\boldsymbol{k}_1\cdots\boldsymbol{k}_n$ for numerals $\boldsymbol{k}_1,\ldots,\boldsymbol{k}_n$ evaluates to a numeral $\boldsymbol{m}$ iff $f(k_1,\ldots,k_n)$ is defined and equals m.*

Thus, computation theory in the form of recursion theory is individualwise represented in all combinatory algebras, however, not structurally: By the generalization of Rice's theorem, there can be no combinator Z which characterizes the set of numerals, nor one which characterizes recursive functions, etc. As is well known, the basic impossibility theorems of recursion theory follow this way.

The principle of combinatory abstraction does not express a uniqueness property; indeed, there are many combinators which satisfy the same defining equation. For example, $\boldsymbol{S} \cdot \boldsymbol{K} \cdot \boldsymbol{K}$ and $\boldsymbol{S} \cdot \boldsymbol{K} \cdot (\boldsymbol{K} \cdot \boldsymbol{K})$ satisfy the same definition as $\boldsymbol{I}$, $\boldsymbol{I} \cdot x = x$. In certain combinatory algebras, called *combinatory models* there is a combinator which singles one particular combinator out of the class of combinators satisfying a defining equation. This *Lambda combinator* $\boldsymbol{L}$ enjoys the following properties:

$$\boldsymbol{L} \cdot x \cdot u = x \cdot u,$$

$$\forall u(X \cdot u = y \cdot u) \Rightarrow \boldsymbol{L}x = \boldsymbol{L}y.$$

It follows that $\boldsymbol{L}x = \boldsymbol{L}y$ iff $\forall u(x \cdot u = y \cdot u)$. Thus $\boldsymbol{L}$ may serve as an eliminator of one universal quantifier from an equation. An immediate generalization extends $\boldsymbol{L}$ to an n-place quantifier-eliminator:

$$\boldsymbol{L}_1 = \boldsymbol{L}, \boldsymbol{L}_{n+1} = (\boldsymbol{BL})(\boldsymbol{BL}_n)$$

for which $\boldsymbol{L}_n \cdot x = \boldsymbol{L}_n \cdot y$ if $\forall u_1, \ldots, u_n(x \cdot u_1 \cdots u_n = y \cdot u_1 \cdots u_n)$. Similarly, the pairing functions $\boldsymbol{P}$, $\boldsymbol{P}_0$, $\boldsymbol{P}_1$ can be composed to yield n-tuples and their components, e.g. triples: $\boldsymbol{T} \cdot u \cdot v \cdot w = \boldsymbol{P}\ (\boldsymbol{P} \cdot u \cdot v) \cdot w$. With these tools it is now possible to prove

Theorem 2.3 Normal form theorem
Let $t_i(a_1, \ldots, a_n, x_1, \ldots, x_m, u_1, \ldots, u_k) = s_i(a_1, \ldots, a_n, x_1, \ldots, x_m, u_1, \ldots, u_k)$, $i = 1, \ldots, p$, be a set of equations with parameters a_j, unknowns x_j and variables u_j. For given a's we need to find x's such that for all u's the above equations hold. Then there exists b and c such that solving the equation $b \cdot x = c \cdot x$ is equivalent to solving the original problem.

The construction is straightforward: First, combinatory abstraction transform each equation to the form $\boldsymbol{T}_i a_1 \ldots u_k = \boldsymbol{S}_i a_1 \ldots u_n$. Next, variables $u_1, \ldots, u_k$ are eliminated in each equation using $\boldsymbol{L}_k$. Then the tupling operators are used to combine the parameters into one, say a, and also the unknowns, say into x. Using tupling again, the resulting equations $t_i'(a, x) = s_i'(a, x)$, $i = 1, \ldots, p$, are combined into one, $t(a, x) = s(a, x)$. This is finally rewritten into $b \cdot x = c \cdot x$ by combinatory abstraction. Each one of these operations is invertible, hence a solution of $b \cdot x = c \cdot x$ can be converted into one of the original problem.

Our favorite example of a combinatory algebra is $\mathcal{D}_A$ for reasons – apart from that of personal attachment – which will become clear as we go along. This algebraic structure is constructed within set theory, starting with a non-empty set A and closing this set by pairing already obtained elements x with finite sets α of such elements. Writing $(\alpha \to x)$ for such pairs, the formal definition is:

$$G_0(A) = A,$$

$$G_{n+1}(A) = G_n(A) \cup \{(\alpha \to x) : \alpha \subseteq F_n(A) \text{ finite}, \, x \in G_n(A)\}.$$

The set of elements of D_A is the set of all subsets of the union $G(A)$ of all $G_n(A)$. This set D_A is made in an algebraic structure by defining the application operation as follows for arbitrary subsets of $G(A)$:

$$M * N = \{x : \exists \alpha \subseteq N \text{ with } (\alpha \to x) \in M\}$$

To verify that $\mathcal{D}_A = \langle D_A, * \rangle$ is a combinatory algebra, it suffices, by Schönfinkel's lemma, to point out two elements in D_A which can fill the roles of $\boldsymbol{S}$ and $\boldsymbol{K}$ respectively:

$$\begin{array}{rcl} K & = & \{(\{y\} \to (\emptyset \to y)) : y \in G(A)\} \\ S & = & \{(\{\tau \to (\{r_1, \ldots, r_n\} \to s)\} \to (\{\sigma_1 \to r_1, \ldots, \sigma_n \to r_n\} \to (\sigma \to s)) : \\ & & n \geq 0, r_1, \ldots, r_n \in G(A), \tau \cup \bigcup \sigma_i = \sigma \subseteq G(A), \sigma \text{ finite } \}. \end{array}$$

The verification of $S * x * y * z = (x * z) * (y * z)$ and $K * x * y = x$ are straightforward.

In fact $\mathcal{D}_A$ is a combinatory model. The lambda-combinator $\boldsymbol{L}$ can be realized by the set $L \subseteq G(A)$, defined by

$$L = \{\{\alpha \to x\} \to (\beta \to x) : \alpha \subseteq \beta \subseteq G(A) \text{ finite}, \, x \in G(A)\}.$$

Again, verification is a simple exercise. (As a general reference for details cf. [Engeler]; constructions similar to $\mathcal{D}_A$ were proposed earlier by Plotkin and Scott, see the classical reference [Barendregt] for combinatory algebra and its relation to the lambda calculus.)

3 An algebraization of universal algebra

A combinatory algebra $\mathcal{D}$ is called γ-universal for an infinite cardinal γ if every combinatory algebra $\mathcal{D}'$ of cardinality $\leq \gamma$ is isomorphic to a substructure of $\mathcal{D}$.

Theorem 3.1 *If $\mid A \mid = \gamma$ then $\mathcal{D}_A$ is γ-universal.*

We actually prove a stronger fact, namely that every algebraic structure of cardinality $\leq \gamma$ having only one binary operation can be isomorphically embedded in $\mathcal{D}$.

The idea of the proof is really quite simple. Consider an algebraic structure with universe A, which is no loss of generality, and a binary operation $\cdot$, written in infix notation. Each element $a \in A$ is viewed as a left multiplier $a \cdot b$ with

b running over A. Inside $\mathcal{D}_A$ this view of a translates into a set of transformation rules which create the representation of the element $c = a \cdot b$ from the representation of b. Formally, this representation is a map $f : A \mapsto D_A$ defined recursively by $f_0(a) = \{a\}, f_{n+1}(a) = f_n(a) \cup \{\{b\} \to x : b \in A, x \in f_n(a \cdot b)\}$; together $f(a) = \bigcup_n f_n(a)$. The map f is one-to-one, since $f(a) \cap A = \{a\}$. One verifies that $f(a) \cdot f(b)$ equals $f(a \cdot b)$ for all a and b by evaluating the definitions.

This proof, and its result, lends itself to generalizations and abstractions from the set-theoretic constructions on which it and the structure $\mathcal{D}_A$ are based. We shall pursue this here also, but with more general algebraic structures and an additional goal: First, the assumption of just one, binary, algebraic operation should be dropped. Second, we should like to describe the universe of the embedded structure in an intrinsic combinatory way.

Therefore, instead of encoding the action of a as a left multiplier into the representation of a itself, consider the notion of multiplication itself encoded into an element m of D_A. This allows us to embed the original structure more easily by $g : A \to D_A$ defined by $g(a) = \{a\}$; then m need only be defined by $m = \{\{a\} \to (\{b\} \to \boldsymbol{c}) : a, b, \boldsymbol{c} \in A\}$. Then g is one-to-one and $g(a \cdot b) = m \cdot g(a) \cdot g(b)$ for all $a, b \in A$. In this fashion each basic operation of a given algebraic structure is embodied in an element of D_A and evaluated as leftmost multiplier in the corresponding subalgebra of $\mathcal{D}_A$.

Indeed, the set of elements of the embedded algebra can also be represented, in a fashion, by a single element of D_A: Consider $r = \{\{a\} \to a : a \in A\} \cup \{\alpha \to x : \emptyset \neq \alpha,\ \alpha$ not of the form $\alpha = \{a\}, a \in A; x \in G(A)\}$. Then $r \cdot \{a\} = \{a\}$ for $a \in A, r \cdot \emptyset = \emptyset, r \cdot x \in G(A)$ for all other $x \in G(A)$. Also, $r \cdot (r \cdot v) = v$ for all v, that is, r is a *retraction.* The *retract* of r, which is the set of all fixpoints of r, forms a complete lattice of subsets in this case, with bottom $\emptyset$ and top $G(A)$.

Altogether then, the given algebraic structure is represented by a list of elements : r, the retraction representing its universe, and an element m for each one of its basic operations. The set m proposed above is almost the correct one (there is some need to accommodate $\emptyset$ and D_A), but leaving such technicalities aside, we have provided what amounts to an *inner algebra* of D_A isomorphic to the original algebra. Formally: Let $\cdot$ be a binary operation on the nonempty set A; extend this algebraic structure by two elements $\top, \bot$ to is completion by defining $x \cdot \bot = \top \cdot x = \bot$ all $x, s \cdot y = \top$ all other cases. Then this structure is isomorphic to the structure obtained on the retract of r using left-multiplication with m.

This formulation of the embedding result generalizes to arbitrary many operations and allows us to consider classes of algebraic structures as tuples of elements of $\mathcal{D}_A$; indeed, by using the tupling operator available in $\mathcal{D}_A$, simply as sets of elements in $\mathcal{D}_A$. This is what is meant by the title of the present chapter.

One problem arises immediately: While $\mathcal{D}_A$ may provide for very many such inner algebras, due to its set-theoretic richness, this is not to be expected of every combinatory model $\mathcal{D}$. The question as to the exact additional axioms has been dealt with by Trudy Weibel, (see chapter II). Furthermore, considering for example classes $\mathcal{V}$ of algebras as sets V of elements of $\mathcal{D}$, the question arises of whether the way that $\mathcal{V}$ was defined – e.g. by a set of equations – can be translated into a way to define V, perhaps again as a set of solutions to equations in $\mathcal{D}$. Again, Weibel was able to lift the positive answer from $\mathcal{D}_A$ to her axiomatically defined class of combinatory models.

By this approach, universal algebra has been turned into a study of solutions of equations in combinatory models, and programmatic questions come to the foreground: What happens to the basic results on universal algebras if they are translated into theorems about combinatory models? This very tempting question has been pursued by Beatrice Amrhein. She shows (see chapter III) that some of these results turn into characterization theorems for solution sets of combinatory equations and into basis theorems that show how to compose general solutions from "irreducible" ones. Finally, Oliver Gloor shows how a theory of extensions of combinatory algebras can be stated and related, for example, to degrees of recursive enumerability.

4 Objects reflected in their properties

For the mathematician, "objects of thoughts" originate either as constructs within mathematical frameworks or as abstractions, again of mathematical character, associated with natural or technological "real" objects. It may be maintained that such objects of thought have reality in so far as they are individuated and that their properties are formulated, or formulable, in a conventionalized language. Our standpoint will be: *objects are to be identified with the totality of their properties.*

As a rule, such formal languages refer to a mathematical framework; e.g. "this body has weight x and temperature y", "this function is periodic of period p and bounded by -2 and $+3$" etc. To be (abstractly) concrete, let us assume that such *basic* properties of objects are formulated in a first-order predicate language. We assume this language to be augmented by a symbol @, intended to denote the object in question in the current context. Thus, the objects above would contain (in their set of properties):

$$\text{weight } @ \ = x \wedge \ \text{temp } (@) \ = y,$$

$$\forall\, t \, [\ (@)\ (t) = \ @\ (t+p) \wedge -2 \leq \ @\ (t) \leq 3].$$

The purpose of theories (of mathematical objects) is to treat of relations between such objects, relations that represent (mathematical or natural) laws. Again, such laws are clear objects of thoughts in so far as they can be individualized and their properties formally described and collected into a set, which we would then again identify with this "law-object". If, for example, we have two pendula p_1 and p_2 coupled by a spring, and we wish to describe the lawful behavior of this system, our approach would be as follows: p_1 and p_2 are sets of formulas (describing the individual behavior of the pendulum); the interaction between p_1 and p_2 is then a pair of set-functions γ_1 and γ_2 with

$$p_1 = \gamma_1(p_1, p_2),\ p_2 = \gamma_2(p_1, p_2).$$

The combinatory approach suggests that the laws γ_1 and γ_2 are again objects of thought, i.e. elements of a suitable combinatory algebra. There is one that lies closely at hand, namely $\mathcal{D}_A$, with $A =$ the set of formulas with which we describe basic properties. Thus we would postulate elements $g_1, g_2 \in \mathcal{D}_A$ with

$$p_1 = g_1 \cdot p_1 \cdot p_2, \quad p_2 = g_2 \cdot p_1 \cdot p_2,$$

indeed, $g_i = \{\alpha \to (\beta \to a) : a \in \gamma_i(\alpha, \beta),\ \alpha \subseteq p_1, \beta \subseteq p_2\}$. In fact, such g_i exist if and only if the γ_i are continuous, i.e. whenever

$$\gamma_i(p, q) = \bigcup\{\gamma_i(\alpha, \beta) : \alpha \subseteq p, \beta \subseteq q,\ \text{finite}\ \}.$$

Of course, one may think of laws γ which do not have this continuity property, but such examples are not easy to come by: in most natural contexts, knowledge about individual objects p is composed of items on whose presence the laws turns; more knowledge about the objects to which the laws apply will result in more knowledge about the effects of the laws.

In chapter III and IV the approach sketched above is completed in three separate types of context:

(a) As a general discipline of modeling systems from the natural and technical sciences by sets of combinatory equations in suitable $\mathcal{D}_A$, (Schwärzler).

(b) As an algebraic theory of approximate (numerical) solutions of functional and differential equations in real analysis, (Aberer).

(c) As a motivation to include among the formal solution to such equations also object-functions which are given by programs in some programming languages, (von Mohrenschildt).

In each of these worked-out examples, the combinatory approach is essential, not only as a device for the presentation of the results, but more importantly as a means of connecting the theory with computing: the approach is always pushed to the point where actual implementation is available: a modeling system CULP (cumulative logic programs) for (a); a system for approximating solutions of differential and functional equations by intervals of rational functions, for (b); and an extension of the Risch algorithms for closed-form integration to discontinuous functions in MAPLE, for (c).

References

[Curry] H. B. Curry, Grundlagen der kombinatorischen Logik. Amer. J. Math. 52 (1930), 509–536, 789–834.

[Dedekind] R. Dedekind, Was sind und was sollen die Zahlen? In: Gesammelte Werke, Bd. 3, 335–391.

[Schönfinkel] M. Schönfinkel, Über die Bausteine der mathematischen Logik. Mathem. Annalen 92 (1924), 305–316.

[Rice] H. G. Rice, Classes of recursively enumerable sets and their decision problems. Trans. Am. Math. Soc. 74 (1953), 358–366.

[Church] A. Church, The Calculi of Lambda-Conversion. Princeton University Press 1941.

[Kleene] S. C. Kleene, λ-definability and recursiveness. Duke Math. J. 2 (1936), 344–353.

[Engeler] E. Engeler, Foundations of Mathematics. Springer Verlag 1993.