(18.145) MATH FROM THE CONSTRUCTIVE PT. OF VIEW.

A GENERAL LANGUAGE

by

Errett Bishop

It would be useful to have a formal language for constructive mathematics comparable in generality to a language such as Zermelo-Skolem-Fraenkel for classical mathematics, as a point of departure for a programming language. The language should permit inductive arguments of some generality. It is immediately clear that it will differ from say ZSF in at least three important respects (in addition to not having an axiom of the excluded middle). First, $t_1 \in t_2$ will be a formula only when the terms t_1 and t_2 are related in a very special way, since it does not make constructive sense to think of the object $S(t_1)$ represented by t_1 as possibly belonging to the object $S(t_2)$ represented by t_2, except when $S(t_1)$ is an element of a certain set X and $S(t_2)$ is a subset of X. Second, there is no reason to prohibit consideration of large sets, such as the set Ω of all sets. Whenever we are able to operate constructively with such large sets (for example, by defining the function f from Ω to Ω by $F(X) \equiv X \times X$), such operations should be represented in the language. Third, the notion of equality must be defined separately for each set, and the scope of the notion does not extend beyond the individual set.

Here we present a tentative version of such a language. The informal usage of [1] was used as a point of departure. The language would seem to be adequate to the routine and natural formalization of all of [1]. The definition of the language is highly recursive: the basic structures--constants, variables, terms, formulas, and proofs--are constructed simultaneously. Each constant, variable, term, or formula is a symbol string (briefly, a string), the symbols being chosen

from an alphabet which we do not specify completely, but which contains the upper and lower case Latin and Greek letters, the digits, the punctuation marks, and certain special mathematical symbols such as $\wedge$, $\vee$, $\rightarrow$, $\forall$, $\exists$, $=$, and $\approx$. (The symbol $\equiv$ does not belong to the alphabet. It is reserved for informal usage; the notation "A $\equiv$ B" means that the string denoted by "A" and the string denoted by "B" are equal.)

Certain strings are called variables, and others are called constants. The basic constants are 0, N, and C, denoting respectively the integer 0, the class of nonnegative integers, and the class of all classes. (A class is a set without an equality relation. In other words, a set is a class together with an equality relation on that class.)

Certain symbol strings are called terms. Each term t represents a mathematical object, which we denote by S(t). Each term t has a type, which is also a term, that we denote by T(t). The type T(T(t)) of the type T(t) of a term t is always C. (A term t of type C represents a class, and a term of type t represents an element of that class.) The types of the basic constants are $T(0) \equiv N$, $T(N) \equiv C$, and $T(C) \equiv C$. If x is any string that otherwise has no meaning, we may declare x to be a variable of type t (provided the term t has type C) by writing $T(x) \equiv t$. Then x itself is a term, of type t. The intuitive meaning is that x ranges over the class represented by t. Each term t represents, rather than denotes, an object; for t to denote an object, values must be assigned to all variables on which t depends (a concept to be explained later). For example, if we declare $T(x) \equiv C$, $T(y) \equiv x$, and $T(z) \equiv Op(x, x)$ (where Op(x, x) represents the class of all operators from the class represented by x to itself), then (z; y) is a term (representing the value of the operator at the element of its domain), and before

$(z; y)$ denotes a definite mathematical object a definite class must be assigned as the value of x, a definite element of that class as the value of y, and a definite operator from that class to itself as the value of z.

If $\exists x A$ is a theorem, and c any string that is otherwise meaningless, we may declare c to represent the entity whose existence is asserted by the theorem and whose construction is given by the proof of the theorem. The type $T(c)$ of c is $T(x)$.

The following rules describe in detail the formation of terms. We use symbols t, t_1, t_2, t', etc. to stand for arbitrary terms, and x, y, z, x_1, x', etc. to stand for arbitrary variables. To help describe the meaning of the language, for each term t we denote the mathematical object represented by t by $S(t)$.

(1t) Each basic or declared constant is a term, whose type has already been described.

(2t) Each declared variable is a term, whose type has already been described.

(3t) If $T(t) \equiv N$, then t^+ is a term of type N (representing the integer $S(t) + 1$).

(4t) If $T(t) \equiv C$, then $P(t)$ is a term of type C (representing the class of all subclasses of $S(t)$). If $T(t_1) \equiv P(t)$, then $|t_1|$ is a term of type C (representing the subclass $S(t_1)$ of $S(t)$ when considered as an abstract class). If $T(t_2) \equiv |t_1|$ then $t_2\downarrow$ is a term of type t (representing the element of $S(t)$ corresponding to the element $S(t_2)$ of $S(|t_1|)$). If $T(t_3) \equiv t$, and $t_3 \in t_1$ is a theorem, then $t_3 \uparrow t_1$ is a term of type $|t_1|$ (representing the element of $S(|t_1|)$ corresponding to the element $S(t_3)$ of $S(t)$).

(5t) If $T(t_1, t_2) \equiv C$ (an abbreviation for $T(t_1) \equiv C$ and $T(t_2) \equiv C$), then $Op(t_1, t_2)$ is a term of type C (representing the class of all operators from $S(t_1)$ to $S(t_2)$).

(6t) If $T(t_1) \equiv Op(t, C)$, then $Op(t : t_1)$ is a term of type C (representing the class of all operators associating to each element α of $S(t)$ an element of the image of α under the operator $S(t_1)$. Such operators will be called guided operators.)

(7t) If $T(t_3) \equiv Op(t_1, t_2)$ and $T(t_4) \equiv t_1$, then $(t_3; t_4)$ is a term of type t_2 (representing the image of $S(t_4)$ under the operator $S(t_3)$).

(8t) If $T(t_1) \equiv Op(t, C)$, $T(t_2) \equiv t$, and $T(t_3) \equiv Op(t : t_1)$, then $(t_3; t_2)$ is a term of type $(t_1; t_2)$ (representing the image of $S(t_2)$ under the guided operator $S(t_3)$).

(9t) If x is final in $dep(x, t)$ (these concepts will be explained later), then $\lambda x; t;$ is a term of type $Op(T(x), T(t))$ (representing the operator whose value at x is t).

(10t) If $T(t_1) \equiv Op(t, C)$, if $T(x) \equiv t$, if $T(t_2) \equiv (t_1; x)$, and if x is final in $dep(x, t_2)$, then $\lambda x; t_2;$ is a term of type $Op(t : t_1)$ (representing the guided operator whose value at x is $S(t_2)$).

(11t) If $T(u) \equiv N$ and $T(t_0, \ldots, t_n) \equiv t$, then $[u : t_0, \ldots, t_n]$ is a term of type t (representing the element $S(t_j)$ of $S(t)$, where $j \equiv \min\{S(u), n\}$).

(12t) If $T(t_1, \ldots, t_n) \equiv C$, then $t_1 \cup \ldots \cup t_n$ is a term of type C (corresponding to the disjoint or exterior union of $S(t_1), \ldots, S(t_n)$). If $T(t_i') \equiv t_i$, then $[t_i', i, t_1 \cup \ldots \cup t_n]$ is a term of type $t_1 \cup \ldots \cup t_n$ (representing the element of $S(t_1 \cup \ldots \cup t_n)$ corresponding

to the element $S(t_i')$ of $S(t_j)$). If t_0 is a term of type $t_1 \cup \ldots \cup t_n$, then $t_0!$ is a term of type N (representing the index j such that $S(t_0)$ arises from an element of the class $S(t_j)$), and $(t_0; i)$, for $1 \leq i \leq n$, is a term of type t_i (representing, in case $i = j$, the element of $S(t_j)$ giving rise to $S(t_0)$).

(13t) If $T(x) \equiv t$, and A is a formula, and if x is final in dep(x, A), then $\{x : A\}$ is a term of type P(t) (representing the class of all x in S(t) that satisfy S(A)).

(14t) Let the distinct variables $x_1, \ldots, x_n$ be a final segment of $dep(x_1, \ldots, x_n)$ (a concept to be explained later), such that x_i does not depend on x_j for $i < j$. Then $G(x_1, \ldots, x_n)$ is a term of type C, called a layered product. (It represents the class of all n-tuples $(a_1, \ldots, a_n)$, where a_1 is obtained by assigning a value to x_1, then a_2 is obtained by assigning a value to a variable x_2' ranging over the class to which $T(x_2)$ specializes due to the assignment of the value a_1 to x_1, and so forth. For example, if $T(x_1) \equiv C$, $T(x_2) \equiv x_1$, $T(x_3) \equiv Op(x_1, C)$, and $T(x_4) \equiv (x_3; x_2)$, then to construct an element of $S(G(x_1, x_2, x_3, x_4))$ we must first construct a class a_1, then an element a_2 of a_1, then an operator a_3 from a_1 to C, and finally an element of the class $a_3(a_2)$). If $x_1, \ldots, x_n \Rightarrow t_1, \ldots, t_n$ is a specialization (a concept to be explained later), then $\langle t_1, \ldots, t_n : G(x_1, \ldots, x_n)\rangle$ is a term of type $G(x_1, \ldots, x_n)$ (representing the n-tuple $(S(t_1), \ldots, S(t_n))$). If $T(t) \equiv G(x_1, \ldots, x_n)$, then for $1 \leq i \leq n$ the strings $(t; i)$ are terms, whose types are determined by the requirement that $x_1, \ldots, x_n \Rightarrow (t; 1), \ldots, (t; n)$ be a specialization. (The term $(t; i)$ represents the i^{th} component of the n-tuple S(t)).

(15t) If $T(x, t_1, t_2) \equiv N$, $T(y, t_3) \equiv t$, $T(t_4) \equiv Op(G(x, y), t)$, and if x and y are independent variables, then $ind(t_1, t_2, t_3, t_4)$ is a term of type t (representing, in case $S(t_1) \leq S(t_2)$, the value when $n = S(t_2)$ of the element $\alpha(n)$ of $S(t)$ determined from the recursion equations $\alpha(S(t_1)) = S(t_3)$, $\alpha(n + 1) \equiv S(t_4)((n, \alpha(n))))$.

We now give the rules for formation of formulas.

(1F) If t_1 and t_2 are terms of type N, then $t_1 = t_2$ and $t_1 \neq t_2$ are formulas.

(2F) If $T(t_1) \equiv t_2$ and $T(t_3) \equiv P(t_2)$, then $t_1 \in t_3$ is a formula.

(3F) If A and B are formulas, so are $A \wedge B$, $A \vee B$, and $A \rightarrow B$.

(4F) If A is a formula and x is final in $dep(x, A)$, then $\forall xA$ and $\exists xA$ are formulas.

A formula represents a mathematical statement, in that when values are assigned to all variables on which it depends, it denotes a definite mathematical statement. When standing alone a formula will be given the generality interpretation, that the corresponding assertion holds for all values of the variables on which it depends.

Our next task is to define what it means for a term t or a formula A to depend on a variable x. For each of the rules (1t) - (15t) and (1F) - (4F), the term or formula being constructed has certain parents, and certain variables are dummied. The complete definitions follow. Unless otherwise stated, no variables are dummied for the given term or formula.

(1at) The basic constants have no parent. The term c has every formula occuring in the proof of $\exists xA$ as a parent, including the formula $\exists xA$ itself.

(2at) The (only) parent of a variable x is $T(x)$.

(3at) The parent of t^+ is t.

(4at) The parent of $P(t)$ is t, of $|t_1|$ is t_1, and of $t_2\downarrow$ is t_2. The parents of $t_3 \uparrow t_1$ are t_3 and t_1.

(5at) The parents of $Op(t_1, t_2)$ are t_1 and t_2.

(6at) The parents of $Op(t : t_1)$ are t and t_1.

(7at) The parents of $(t_3; t_4)$ are t_3 and t_4.

(8at) The parents of $(t_3; t_2)$ are t_2 and t_3.

(9at) The parents of $\lambda x; t;$ are x and t, and x is dummied.

(10at) The parents of $\lambda x; t_2;$ are x and t_2, and x is dummied.

(11at) The parents of $[u : t_0, \ldots, t_n]$ are u, t_0, $\ldots$, and t_n.

(12at) The parents of $t_1 \cup \ldots \cup t_n$ are $t_1, \ldots, t_n$. The parents of $[t_i', i, t_1 \cup \ldots \cup t_n]$ are t_i' and $t_1 \cup \ldots \cup t_n$. The parent of $t_0!$ and $(t_0; i)$ is t_0.

(13at) The parents of $\{x : A\}$ are x and A, and x is dummied.

(14at) The parents of $G(x_1, \ldots, x_n)$ are $x_1, \ldots, x_n$, and $x_1, \ldots, x_n$ are dummied.

(15at) The parents of $\mathrm{ind}(t_1, t_2, t_3, t_4)$ are t_1, t_2, t_3, and t_4.

(1aF) The parents of $t_1 = t_2$ and $t_1 \neq t_2$ are t_1 and t_2.

(2aF) the parents of $t_1 \in t_3$ are t_1 and t_3.

(3aF) The parents of $A \wedge B$, $A \vee B$, and $A \to B$ are A and B.

(4aF) The parents of $\forall xA$ and $\exists xA$ are x and A, and x is dummied.

We now define dependence.

(3D) Every variable x depends on itself. If x is not dummied for the term t (respectively, formula A), and some parent of t (respectively, A) depends on x, then t (respectively, A) depends on x.

We shall use the notation $x \leq t$ to mean that the term t depends on the variable x. A fundamental property of the relation $\leq$ (which we shall not need) is that if $x \leq T(t)$ then $x \leq t$. (The proof, by induction, is left to the reader.) Another fundamental property is that if $x \leq y$ and $y \leq t$ then $x \leq t$. To see this, assume $x \leq y$ and $y \leq t$. Then $y \leq t'$ for some parent t' of t. By the inductive hypothesis, $x \leq t'$. Since y is not dummied for t, neither is the variable x on which it depends (as can be seen by checking the definitions of which variables are dummied for the various rules). Hence, $x \leq t$. Another fundamental property is that if $x \leq y$ and $y \leq x$, then x and y are the same variable. This is proved by induction on the number n of variables currently

declared: it is true when $n = 0$, and if it is true for a given n, and n is increased by 1 by the declaration $T(x) \equiv t$, then the relation $x \leq y$ fails to hold for all variables y previously declared. Thus $\leq$ is a partial ordering on the set of all declared variables.

If $t_1, \ldots, t_m$ are terms and $A_1, \ldots, A_n$ are formulas, $dep(t_1, \ldots, t_m, A_1, \ldots, A_n)$ is the set of all variables on which one of the terms or formulas depends. (It is an initial segment with respect to $\leq$, in the sense that if y belongs to it and $x \leq y$, then so does x.) An element x of such an initial segment S is called final if $y \in S$ and $x \leq y$ imply that $y \equiv x$. A subset S_1 of S is called a final segment of S if $y \in S$ and $x \leq y$ for some x in S_1 imply that $y \in S_1$. The variables $x_1, \ldots, x_n$ are independent if none of them depends on any of the others.

We next define what it means to specialize a finite sequence $z_1, \ldots, z_k$ of variables. The requirement on the sequence is that $z_1, \ldots, z_k$ be a final segment of $dep(z_1, \ldots, z_k)$, and that z_i not depend on z_j for $i < j$. A specialization consists in assigning to each of the variables z_i a term v_i, which is indicated by $z_1, \ldots, z_k \Rightarrow v_1, \ldots, v_k$, such that $T(v_1) \equiv T(z_1)$ and

$$T(v_i) \equiv im(T(z_i) : z_1, \ldots, z_{i-1} \Rightarrow v_1, \ldots, v_{i-1})$$

for $2 \leq i \leq k$, where $im(t : z_1, \ldots, z_k \Rightarrow v_1, \ldots, v_k)$ stands for a certain term, the image of the term t with respect to the given specialization, which we now define. To do so, we fix the specialization $z_1, \ldots, z_k \Rightarrow v_1, \ldots, v_k$, and let $im(t)$ (respectively $im(A)$) stand for the image of a term t (respectively a formula A) with respect to that fixed specialization. For $im(t)$ (respectively $im(A)$) to be defined, it is required that for each variable u belonging to $dep(z_1, \ldots, z_k, t)$ (respectively $dep(z_1, \ldots, z_k, A)$), either $u \equiv z_i$ for some i or else $im(T(u))$ is defined and $im(T(u)) \equiv T(u)$.

In case t is a basic constant, $im(t) \equiv t$. In case the constant c is declared to represent the element constructed by the proof P of the formula $\exists xB$, then $im(c)$ is a constant declared to represent the element constructed by the proof $im(P)$ of the formula $im(\exists xB)$ (the proof $im(P)$ will be defined later). In case x is a variable, then $im(x) \equiv v_i$ in case $x \equiv z_i$ and $im(x) \equiv x$ otherwise. For a term t that is neither a variable nor a constant, or for a formula A, let $t_1, \ldots, t_m, A_1, \ldots, A_n$ be the parents of t (respectively A), and let $x_1, \ldots, x_p$ be dummied. We may assume, after reordering if necessary that x_i does not depend on x_j for $i < j$. Let $x'_1, \ldots, x'_p$ be newly declared variables, with $T(x'_1) \equiv im(T(x_1))$ and $T(x'_i) \equiv im(T(x_i) : \overline{z_1}, \ldots, \overline{z_k}, x_1, \ldots, x_{i-1} \Rightarrow \overline{v_1}, \ldots, \overline{v_k}, x'_1, \ldots, x'_{i-1})$ for $2 \leq i \leq p$, where $\overline{z_i}$ stands for z_i and $\overline{v_i}$ for v_i, except that they are left out of the list if $x_j \leq z_i$ for some $j (1 \leq j \leq p)$. Let t'_i (respectively A'_i) be the image of t_i (respectively A_i) with respect to the specialization

$$z_1, \ldots, z_k, x_1, \ldots, x_p \Rightarrow v_1, \ldots, v_k, x'_1, \ldots, x'_p.$$

Then $im(t)$ (respectively $im(A)$) is obtained from t (respectively A) by replacing each of its parents t_i or A_i by its image t'_i or A'_i. (For example, $im(Op(t, t')) \equiv Op(im(t), im(t'))$ and $im(\{x_1 : A\}) \equiv \{x'_1 : A'\}$, etc.)

Finally, if P is a proof of a theorem A, the proof $im(P)$ of $im(A)$ is obtained as follows. If A is an axiom, $im(A)$ will have a canonical proof. If A is the conclusion of some rule of inference, whose premises are $A_1, \ldots, A_n$, then there exist canonical specializations $\sigma_1, \ldots, \sigma_n$, uniquely determined by the specialization $z_1, \ldots, z_k \Rightarrow v_1, \ldots, v_k$ and the number of the rule of inference, and a canonical deduction D of $im(A)$ from $im(A_1 : \sigma_1), \ldots, im(A_n : \sigma_n)$. The proof $im(P)$ then consists in proving

$\mathrm{im}(A_i : \sigma_i)$ by the proof $\mathrm{im}(P_i : \sigma_i)$ (where P_i is the proof of A_i) for $1 \le i \le n$, and then deducing $\mathrm{im}(A)$ by D.

We next give the axioms and rules of inference of our language. The first three axioms and seven rules are grouped together, since they are customarily called the axioms and rules for the (constructive) propositional calculus. The symbols A, B, and C stand for arbitrary formulas, so that each "axiom" as given develops into an infinity of actual axioms, obtained by substituting arbitrary formulas for A, B, and C. The same is true for each "rule".

(1A) $A \to A$.

(1R) If B, then $A \to B$.

(2R) If A and $A \to B$, then $(\exists x_1 \ldots \exists x_n\ 0 = 0) \to B$ where $x_1, \ldots, x_n$ are the distinct variables in appropriate order of $\mathrm{dep}(A) - \mathrm{dep}(B)$.

(3R) If $A \to B$ and $B \to C$, then $(\exists x_1 \ldots \exists x_n\ 0 = 0) \to (A \to C)$ where $x_1, \ldots, x_n$ are the distinct variables in appropriate order of $\mathrm{dep}(B) - \mathrm{dep}(A, C)$.

(2A) $A \wedge B \to A$, $B \wedge A \to A$, $A \to A \vee B$, $B \to A \vee B$.

(4R) If $A \to C$ and $B \to C$ then $A \vee B \to C$.

(5R) If $C \to A$ and $C \to B$ then $C \to A \wedge B$.

(6R) If $A \to (B \to C)$ then $(A \wedge B) \to C$.

(7R) If $(A \wedge B) \to C$ then $A \to (B \to C)$.

(3A) $0 = 0^+ \to A$.

The following two axioms and two rules, together with those already given, complete the description of the constructive predicate calculus. In the next two rules, A does not depend on the variable x, and x is final in $\mathrm{dep}(x, B)$.

(8R) If $(\exists x\ 0 = 0) \to (A \to B)$, then $A \to \forall x B$.

(9R) If $(\exists x\ 0 = 0) \to (B \to A)$, then $\exists x B \to A$.

In the next two axioms, x is assumed to be final in $dep(x, A)$. We let $A(x)$ stand for A, and $A(t)$ stand for $im(A : x \Rightarrow t)$.

(4A) $K \to (\forall x A(x) \to A(t))$.

(5A) $K \to (A(t) \to \exists x A(x))$,

where $K \equiv 0 = 0$ if A depends on x, and $K \equiv \exists x(0 = 0)$ if not.

The next axiom is the axiom of choice. The string $\forall x \exists y A$ must be a formula, y must not depend on x, and $T(z) \equiv Op(T(x), T(y))$. As above, $A(x, y)$ stands for A, and $A(x, (z; x))$ stands for $im(A : y \Rightarrow (z; x))$.

(6A) $\forall x \exists y A(x, y) \to \exists z \forall x A(x, (z; x))$.

The same axiom also holds under the following circumstances: there is a term t of type $Op(T(x), C)$, not depending on x, with $T(y) \equiv (t; x)$ and $T(z) \equiv Op(T(x) : t)$.

The next rule is mathematical induction.

(10R) If $A(0)$ and $A(x) \to A(x^+)$, then $A(x)$.

The next rule concerns the declared constants.

(11R) If $\exists x A(x)$, then $A(c)$, where c is the constant representing the quantity constructed in the proof of $\exists x A(x)$.

The next axiom concerns rule (13t).

(7A) $t_1 \in \{x : A(x)\} \leftrightarrow A(t_1)$ (where $B \leftrightarrow C$ stands for $(B \to C) \wedge (C \to B)$).

The following axioms describe the order properties of the integers. All terms have type N.

(8A) $t_1 = t_2 \vee t_1 \neq t_2$.

(9A) $(t_1 = t_2) \wedge (t_1 \neq t_2) \to 0 = 0^+$.

(10A) $t_1 = t_2 \to t_1^+ = t_2^+$ $\qquad$ $t \neq t^+$.

In case the variables $x_1, \ldots, x_n$ are independent, $G(x_1, \ldots, x_n)$ is what is usually called the cartesian product of $T(x_1), \ldots, T(x_n)$. To get closer to the usual notation for cartesian products, consider terms $t_1, \ldots, t_n$ of type C. We presume we have fixed some algorithm which to each such finite sequence declares a canonical sequence $x_1, \ldots, x_n$ of independent variables, with $T(x_i) \equiv t_i$. We then define

$$t_1 \times \ldots \times t_n \equiv G(x_1, \ldots, x_n).$$

Also, for arbitrary terms $t_1', \ldots, t_n'$, with $T(t_i') \equiv t_i$, we abbreviate the element $\langle t_1', \ldots, t_n' : t_1 \times \ldots \times t_n \rangle$ by $\langle t_1', \ldots, t_n' \rangle$, with no ambiguity.

The next axiom uses the notation of (4t).

(11A) $$t_2\downarrow \in t_1.$$

Before giving the remaining axioms, we introduce a special notion, which expresses the fact that two elements of a given class are the same mathematical object, in the sense that one belongs to an arbitrary subclass if and only if the other does too.

(4D) Let $T(t) \equiv C$, $T(t_1, t_2) \equiv t$, and $T(x) \equiv P(t)$. Then $t_1 \approx t_2$ stands for the formula

$$\forall x(t_1 \in x \leftrightarrow t_2 \in x).$$

The next axiom uses the notation of (4t)

(12A) $t_2 \downarrow\uparrow t_1 \approx t_2$, $(t_3 \uparrow t_1)\downarrow \approx t_3$.

The next axiom uses the notation of (9t). We assume $T(t') \equiv T(x)$.

(13A) $(\lambda x;\ t(x);;\ t') \approx t(t')$,

where $t(x)$ stands for t, and $t(t')$ stands for $\mathrm{im}(t : x \Rightarrow t')$.

The next axiom uses the notation of (10t). We assume $T(t') \equiv T(x)$.

(14A) $(\lambda x;\ t_2(x);;\ t') \approx t_2(t')$.

The next axiom uses the notation of (11t). The expression $0^{(i)}$ stands for the term $0^{+\ldots+}$, with i plus signs.

(15A) $$u = 0^{(i)} \rightarrow [u : t_0, \ldots, t_n] \approx t_j,$$

where $j \equiv \min\{i, n\}$,

$$u \neq 0^{(0)} \wedge \ldots \wedge u \neq 0^{(n)} \rightarrow [u : t_0, \ldots, t_n] \approx t_n$$

The next axiom uses the notation of (12t).

(16A) $$[t_i', i, t_1 \cup \ldots \cup t_n]! = 0^{(i)},$$

$$([t_i', i, t_1 \cup \ldots \cup t_n];\ i) \approx t_i',$$

$$t'! = 0^{(i)} \rightarrow [(t';\ i), i, t_1 \cup \ldots \cup t_n] \approx t'.$$

The next two axioms use the notation of (14t).

(17A) $\langle (t; 1), \ldots, (t; n) : G(x_1, \ldots, x_n) \rangle \approx t$

(18A) $\langle t_1, \ldots, t_n :\rangle \approx \langle t_1', \ldots, t_n' :\rangle \to (A(t_1, \ldots, t_n) \leftrightarrow A(t_1', \ldots, t_n'))$,

where $A(x_1, \ldots, x_n)$ is any formula such that $A(t_1, \ldots, t_n)$ and $A(t_1', \ldots, t_n')$ are defined, and $\langle t_1, \ldots, t_n :\rangle$ stands for $\langle t_1, \ldots, t_n : G(x_1, \ldots, x_n) \rangle$.

The next axiom expresses a special property of the integers. We assume $T(t_1, t_2) \equiv N$.

(19A) $t_1 = t_2 \to t_1 \approx t_2$.

The next axiom uses the notation of (16t).

(20A) $t_1 = t_2 \to \mathrm{ind}(t_1, t_2, t_3, t_4) \approx t_3$.

$t_2 = t_1^{(i)} \to (\mathrm{ind}(t_1, t_2, t_3, t_4) \approx \mathrm{ind}(t_1^+, t_2, (t_4; \langle t_1, t_3 \rangle), t_4)$,

for each positive integer i.

It should be stressed that certain seemingly artificial restrictions of the language are motivated by the desire not to press the search for constructive meaning too far. This does not mean that we are primarily concerned with avoiding contradictions. A contradiction would be just an indication that we were indulging in meaningless formalism.

From the classical point of view, one could probably strengthen the language considerably, beyond adding the axiom of the excluded middle and some version of the axiom of choice, but this does not concern us.

Next, we consider how to develop some standard mathematical concepts within the language. For convenience, we let $\{z \in t : A\}$ stand for $\{z : A\}$, $\forall z \in t\, A$ stand for $\forall zA$, and $\exists z \in t\, A$ stand for $\exists zA$, where $T(z) \equiv t$. We use t^n to stand for $t \times \ldots \times t$, with n factors.

Our first task is to formalize the concept of a set, as a layered product consisting of a class together with an equality relation on that class. This is done by the following two definitions.

(1D) Let $T(t) \equiv C$ and $T(x_1, x_2, x_3) \equiv t$. We define

$$E(t) \equiv \{x \in P(t^2) : \forall x_1 \forall x_2 \forall x_3\, (\langle x_1, x_1\rangle \in x \wedge (\langle x_1, x_2\rangle \in x \rightarrow \langle x_2, x_1\rangle \in x) \wedge ((\langle x_1, x_2\rangle \in x \wedge \langle x_2, x_3\rangle \in x) \rightarrow \langle x_1, x_3\rangle \in x))\}.$$

(Of course, $E(t)$ represents the class of all equality relations on $S(t)$.)

(2D) Let $T(x) \equiv C$ and $T(y) \equiv E(x)$. Then $U \equiv G(x, y)$.

(Of course, U represents the class of all sets.)

To get closer to the usual notation for sets, consider a term t of type U. (Then $(t; 1)$ represents the class underlying $S(t)$ and $(t; 2)$ represents the subclass of $S((t; 1))^2$ that defines the equality relation on $S((t; 1))$. We shall use the notation $T(t_1) \equiv t$ to mean that $T(t_1) \equiv (t; 1)$, and also to indicate that whenever $T(t_1) \equiv T(t_2) \equiv t$, then $t_1 = t_2$ stands for $\langle t_1, t_2\rangle \in (t; 2)$.

If $T(t_1, \ldots, t_n) \equiv U$, then $t_1 \times \ldots \times t_n$ stands for

$\langle (t_1; 1) \times \ldots \times (t_n; 1), eq : U \rangle$, where

$$eq \equiv \{z \in ((t_1; 1) \times \ldots \times (t_n; 1))^2 :$$
$$((z; 1); 1) = ((z; 2); 1) \wedge \ldots \wedge ((z; 1); n) = ((z; 2); n)\}$$

where $((z; 1); i)$ and $((z; 2); i)$, which have type $(t_i; 1)$, are of course considered to have type t_i. In general, if $T(t) \equiv t_1 \times \ldots \times t_n$, then $(t; i)$ is considered to have type t_i. Thus if $T(t, t') \equiv t_1 \times \ldots \times t_n$, then $t = t'$ stands for $\langle t, t' \rangle \in eq$, so

$$\begin{aligned} t = t' &\leftrightarrow \langle t, t' \rangle \in eq \\ &\leftrightarrow ((\langle t, t' \rangle; 1); 1) = ((\langle t, t' \rangle; 2); 1) \wedge \ldots \\ &\quad \wedge ((\langle t, t' \rangle; 1); n) = ((\langle t, t' \rangle; 2); n) \\ &\leftrightarrow (t; 1) = (t'; 1) \wedge \ldots \wedge (t; n) = (t'; n), \end{aligned}$$

by (18A).

Assume now $T(t_1) \equiv C$ and $T(t_2) \equiv U$. Then $Op(t_1, t_2)$ stands for

$$\langle Op(t_1, (t_2; 1)), eq : U \rangle$$

where

$$eq \equiv \{z \in (Op(t_1, (t_2; 1)))^2 : \forall x \in t_1(((z; 1); x) = ((z; 2); x))\}$$

with $((z; 1); x)$ and $((z; 2); x)$, which have type $(t_2; 1)$, considered to have type t_2. In general, if $T(t) \equiv t_1$ and $T(t') \equiv Op(t_1, t_2)$, then $(t'; t)$ is considered to have type t_2.

In case $T(t_1) \equiv U$ and $T(t_2) \equiv U$ or C, then $Op(t_1, t_2)$ stands for $Op((t_1; 1), t_2)$. If $T(t_1, t_2) \equiv U$, we set

$$F(t_1, t_2) \equiv \{x \in Op(t_1, t_2) : \forall x_1 \in t_1\ \forall x_2 \in t_1(x_1 = x_2 \rightarrow (x; x_1) = (x; x_2))\},$$

where $T(F(t_1, t_2)) \equiv P(Op(t_1, t_2))$, a notation to be explained presently.

Similar conventions can be given for guided operators. This is left to the reader.

If $T(t) \equiv U$, we set

$$P(t) \equiv \langle P((t; 1)), eq : U\rangle,$$

where

$$eq \equiv \{x \in P((t; 1))^2 : \forall x \in t((x \in (z; 1) \rightarrow \exists y \in t(y = x$$

$$\wedge\ y \in (z; 2))) \wedge (x \in (z; 2) \rightarrow \exists y \in t(y = x \wedge y \in (z; 1))))\}.$$

For each t_1 with $T(t_1) \equiv P(t)$, we set

$$|t_1| \equiv \langle |(t_1; 1)|, eq : U\rangle,$$

where

$$eq \equiv \{z \in |(t_1;\ 1)|^2 : (z;\ 1)\downarrow\ = (z;\ 2)\downarrow\ \},$$

with $(z;\ 1)\downarrow$ and $(z;\ 2)\downarrow$, which have type $(t;\ 1)$, considered to have type t.

The term U represents the class of all sets. We can make U into a set U_0 as follows:

$$U_0 \equiv \langle U,\ eq : U\rangle,$$

where

$$eq \equiv \{z \in U^2 : \exists f \in |F((z;\ 1),\ (z;\ 2))|\ \exists g \in |F((z;\ 2),\ (z;\ 1))|$$

$$(\forall x \in (z;\ 1)((g\downarrow;\ (f\downarrow;\ x))\ = x) \wedge \forall y \in (z;\ 2)((f\downarrow;\ (g\downarrow;\ y))\ = y))\}.$$

These examples should suffice to indicate how the conventions for classes, which are built in, can be extended, by appropriate definitions, to work for sets.

We next show how to represent the class of all categories. Write $T(x_1) \equiv C$ and $T(x_2) \equiv Op(x_1 \times x_1,\ U)$. Write $T(y_1,\ y_2,\ y_3) \equiv x_1$, and

$$F \equiv |F((x_2;\ \langle y_1,\ y_2\rangle) \times (x_2;\ \langle y_2,\ y_3\rangle),\ (x_2;\ \langle y_1,\ y_3\rangle))|.$$

Write

$$T(x_3) \equiv Op(x_1 : \lambda y_1;\ Op(x_1 : \lambda y_2;\ Op(x_1 : \lambda y_3;\ F;);););)$$

and

$$T(x_4) \equiv Op(x_1 : \lambda y_1;\ (x_2;\ \langle y_1,\ y_1\rangle);).$$

Then the class of all categories is represented by

$$\mathrm{Cat} \equiv \{\alpha \in G(x_1, x_2, x_3, x_4) : \forall v_1 \forall v_2 \forall v_3 \forall v_4\ A\}$$

where $T(v_1, v_2, v_3, v_4) \equiv \alpha_1$ and the formula A is defined as follows. Let $\alpha_i \equiv (\alpha; i)$, for $1 \leq i \leq 4$. If $T(t_1, t_2, t_3) \equiv \alpha_1$, $T(t_4) \equiv (\alpha_2; \langle t_1, t_2 \rangle)$, and $T(t_5) \equiv (\alpha_2; \langle t_2, t_3 \rangle)$, we define

$$t_5 \circ t_4 \equiv ((((\alpha_3; t_1); t_2); t_3)\downarrow; \langle t_4', t_5 \rangle).$$

Then

$$A \equiv \forall w \in (\alpha_2; \langle v_1, v_2 \rangle)(A_1 \wedge A_2)$$

$$A_1 \equiv w \circ (\alpha_4; v_1) = w \wedge (\alpha_4; v_2) \circ w = w$$

$$A_2 \equiv \forall w' \in (\alpha_2; \langle v_2, v_3 \rangle)\ \forall w'' \in (\alpha_2; \langle v_3, v_4 \rangle)\ A_3$$

$$A_3 \equiv w'' \circ (w' \circ w) = (w'' \circ w') \circ w.$$

As an exercise, the reader should now define the category of groups (as a term of type Cat), and the category of categories and functors.

Since we haven't yet developed the theory of the real numbers, we discuss the set of all Borel subsets of the nonnegative integers, rather than the set of all Borel subsets of the real numbers. Let $N_0 \equiv G(N, \{z \in N^2 : (z; 1) = (z; 2)\}$: For $T(m, n) \equiv N_0$, $T(v, w) \equiv P(N_0)$, $T(x, y) \equiv P(P(N_0))$, and $T(f) \equiv \mathrm{Op}(N_0, |y|)$, we define

$$A \equiv \{x : \forall n(\{n\} \in x) \wedge \forall y((y \subset x \wedge \forall n(\{n\} \in y) \wedge$$

$$\forall f(\mathrm{Un}\ f \in x \to \mathrm{Un}\ f \in y)) \to y = x)\},$$

where

$$\{n\} \equiv \{m : m = n\},$$

$$y \subset x \equiv \forall v(v \in y \to \exists w(v = w \wedge w \in x)),$$

$$\mathrm{Un}\ f \equiv \{m : \exists n(m \in (f; n)\downarrow)\}.$$

Then we define

$$\mathrm{Borel}\ N \equiv \{w : \exists x(x \in A \wedge w \in x)\}.$$

It is easy to derive the characteristic properties of Borel N:

$$\forall n(\{n\} \in \mathrm{Borel}\ N),$$

$$\forall g \in \mathrm{Op}(N, |\mathrm{Borel}\ N|)(\mathrm{Un}\ g \in \mathrm{Borel}\ N)$$

$$\forall y((\forall n(\{n\} \in y) \wedge \forall f(\mathrm{Un}\ f \in y)) \to (\mathrm{Borel}\ N \subset y)).$$

Actually, we have defined Borel sets as ordinary sets, rather than as complemented sets (see [1] for this concept), the more interesting case, but the principle of definition for complemented sets would be the same.

The cardinal numbers, of course, are just the set U_0. The constructive ordinals, as developed by Brouwer [3], can easily be formalized. Since the equality relations for ordinals are rather complicated, and not relevant here, we content ourselves with realizing the ordinals as a class.

Informally, the ordinals are obtained inductively from the following two principles:

(10) every nonnegative integer is an ordinal,

(20) every sequence of ordinals already constructed is an ordinal.

The idea for formalizing this definition is to represent the integer n by the sequence $0, n, \ldots$, where the remaining terms do not matter, and to represent the ordinal α corresponding to the sequence $\alpha_1, \alpha_2, \ldots$ of ordinals by the sequence $k_1, k_2, \ldots$ of integers, where $k_1 \equiv 1$ and $k_2, k_3, \ldots$ is an encoding as a single sequence of the sequence $s_1, s_2, \ldots$, where s_i is the sequence of integers that represents α_i. Let us assume we have already defined a term p of type $\mathrm{Op}(N, N \times N)$ and a term q of type $\mathrm{Op}(N \times N, N)$, such that

$$x \neq 0 \to (q; (p; x)) = x$$

$$(q; y) \neq 0 \wedge (p; (q; y)) = y$$

where $T(x) \equiv N$ and $T(y) \equiv N \times N$. Set $T(\omega) \equiv \mathrm{Op}(N, N)$, $T(A) \equiv P(\mathrm{Op}(N, N))$, and $T(x') \equiv N$, and define

$$\theta \equiv \{A : \forall\omega[((\omega; 0) = 0) \vee$$

$$\forall x \lambda x'; (\omega; (q; \langle x, x' \rangle)); \in A) \to \omega \in A]\},$$

and

$$\mathrm{Ord} \equiv \{\omega : \forall A(A \in \theta \to \omega \in A)\}.$$

It might be helpful to comment on how a general theorem, say about groups, will "specialize" to individual groups. The definition of the term Group, representing the class of all groups, will have the form

$$\text{Group} \equiv \{z \in G(x, y) : A\}.$$

The general theorem about groups, which we wish to "specialize" to the particular group represented by the term $\langle t_1, t_2 :\rangle$, will have the form

$$B \equiv \text{im}(B_0 : x, y, x_1, \ldots, x_n \Rightarrow (z; 1), (z; 2), x_1', \ldots, x_n').$$

If we set

$$B' \equiv \forall x_1' \ldots \forall x_n' B,$$

then B' is also a theorem, and conversely if B' is a theorem, then B is a theorem. Therefore we work with B' rather than B, which means we may assume

$$B \equiv \text{im}(B_0 : x, y \Rightarrow (z; 1), (z; 2)).$$

Then

$$B_1 \equiv \text{im}(B : z \Rightarrow \langle t_1, t_2 :\rangle)$$

$$\equiv \text{im}(B_0 : x, y \Rightarrow (\langle t_1, t_2 :\rangle; 1), (\langle t_1, t_2 :\rangle; 2))$$

is also a theorem. By (18A),

$$B_2 \equiv \text{im}(B_0 : x, y \Rightarrow t_1, t_2)$$

is a theorem, which is the "specialization" of B to the group $\langle t_1, t_2 :\rangle$.

Next, we indicate the power of the equivalence relation $\approx$. Let the terms $\langle t_1, t_2 :\rangle$ and $\langle t_1, t_2' :\rangle$ have type $G(x_1, x_2)$, and let $T(u) \equiv P(T(t_2))$. Since $T(t_2) \equiv \mathrm{im}(T(x_2) : x_1 \Rightarrow t_1) \equiv T(t_2')$, we also have $T(u) \equiv P(T(t_2'))$. Setting $A(x_1, x_2) \equiv x_2 \in u$, from (18A) we get

$$\langle t_1, t_2 :\rangle \approx \langle t_1, t_2' :\rangle \to (t_2 \in u \leftrightarrow t_2' \in u),$$

from which we get

$$\langle t_1, t_2 :\rangle \approx \langle t_1, t_2' :\rangle \to t_2 \approx t_2'.$$

This formula can be considerably generalized, but we shall not do so here.

As an application, let $T(x_1, t_1, t_2) \equiv C$, $T(f) \equiv \mathrm{Op}(x_1, t_2)$, and $T(x_2) \equiv x_1$. Write

$$\mathrm{Equ}(f, x_1, t_2) \equiv \forall x_2(\langle x_1, x_2 :\rangle \approx \langle t_2, (f; x_2) :\rangle),$$

where $\langle x_1, x_2 :\rangle$ stands for $\langle x_1, x_2 : G(x_1, x_2)\rangle$, and similarly for $\langle t_2, (f; x_2) :\rangle$. Set

$$A(x_1) \equiv \exists f\ \mathrm{Equ}(f, x_1, t_2).$$

Then $A(t_2)$ is a theorem, and so is

$$t_1 \approx t_2 \to (A(t_1) \leftrightarrow A(t_2)).$$

Hence $t_1 \approx t_2 \to A(t_1)$, or

$$t_1 \approx t_2 \to \exists g\ \mathrm{Equ}(g, t_1, t_2),$$

where $T(g) \equiv \mathrm{Op}(t_1, t_2)$. Moreover, it is easily seen that

$$\mathrm{Equ}(g, t_1, t_2) \wedge \mathrm{Equ}(g', t_1\ \ t_2) \to \forall x((g; x) \approx (g'; x)),$$

where $T(x) \equiv t_1$, and that

$$\mathrm{Equ}(g, t_1, t_2) \wedge \mathrm{Equ}(h, t_2, t_1) \to \forall x((h; (g; x)) \approx x).$$

Thus any term t such that $\mathrm{Equ}(t, t_1, t_2)$ represents the identity operator from the set $S(t_1)$ (which is the same as the set $S(t_2)$) to itself.

These considerations can be used to formalize the concept of the exterior union of a family of sets. Let $T(t) \equiv C$ and $T(t') \equiv \mathrm{Op}(t, U)$. Write $T(x) \equiv t$ and $T(y) \equiv (t'; x)$. Then $G(x, y)$ represents the exterior union of the sets $S((t'; x))$ over all x in $S(t)$, as a class, and the equality relation on it is represented by the term

$$eq \equiv \{z \in G(x,y)^2 : \exists w \in (t';((z;1);1))(w = ((z;1);2) \wedge \langle((z;1);1), w:\rangle \approx (z;2))\}.$$

The not-quite-trivial proof that $eq \in E(G(x, y))$ is left to the reader. The reader should also show that if $\langle t_1, t_2 :\rangle$ and $\langle t_1', t_2' :\rangle$ are terms of type $\langle G(x, y), eq : U\rangle$, then

$$\langle t_1, t_2 :\rangle = \langle t_1', t_2' :\rangle \to t_1 \approx t_1',$$

$$\langle t_1, t_2 :\rangle = \langle t_1, t_2' :\rangle \to t_2 = t_2',$$

and

$$t_1 \approx t_1' \to \exists z \in (t'; t_1')(\langle t_1, t_2 :\rangle = \langle t_1', z :\rangle).$$

The case of most interest of course is the case $t \equiv N$, corresponding to the exterior union of a sequence of sets.

Now we have left two axioms until now, because they are possibly somewhat controversial. The first axiom generalizes what in [2] was called the axiom of choice. We set $T(t, t') \equiv C$, $T(x) \equiv t$, and $T(t_1, t_2) \equiv Op(t, t')$.

(21A) $$\forall x((t_1; x) \approx (t_2; x)) \to t_1 \approx t_2.$$

This axiom would seem to be no more controversial than the rest of our system; no formalism of such generality can be trusted completely. If aspects of the formalization are meaningless, experience will sooner or later let us know. A similar axiom, whose statement is left to the reader, holds for guided operators. The following axiom concerns the equivalence relation. We set $T(t) \equiv C$, $T(x) \equiv t$, and $T(t_1, t_2) \equiv P(t)$.

(22A) $$\forall x(x \in t_1 \leftrightarrow x \in t_2) \to t_1 \approx t_2.$$

The reader has undoubtedly realized that there are many metatheorems implicit in the constructions of the language. These theorems have not been made explicit, because it is thought that the language is better learned by justifying the constructions intuitively, on the basis of their meanings.

It seems very probable that the language can be compiled, in the sense of [2]. Again, a two (or more) stage compilation would seem to be indicated.

The first stage will be to compile the language into a strictly algorithmic language (one containing only terms-no formulas or proofs). In [2], the language was compiled into the set of constant-free terms. The present language does not readily compile into the set of its constant-free terms, partly because it is not clear what sort of term should compile a proof of a formula of the form $t_1 \in t_2$.

The special interest in compiling the present language into a strictly algorithmic language is that the language seems to encompass most if not all of existent constructive mathematics. Therefore an algorithmic language into which it could naturally be compiled would be extremely general-perhaps too general, perhaps not. In any case, it would be interesting to have such a language, as a standard of comparison for existing programming languages and programming languages under development, and as a point of departure for a very general programming language. It is intended to examine these matters in a subsequent paper.

REFERENCES

[1] E. Bishop, Foundations of Constructive Analysis, McGraw-Hill, New York, 1967.

[2] E. Bishop, How to Compile Certain Formal Languages, to appear.

[3] L. E. J. Brouwer, Zur Begründung der intuitionistischen Mathematic. I-II. Math. Annalen 93, pp. 244-257 (1925); 95, pp. 453-472 (1926); 96, pp. 451-488 (1927)