EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics

# SOME EXTENSIONS OF AUTOMATH : THE AUT-4 FAMILY.

by

N.G. de Bruijn.

January 1974.

University of Technology
Department of Mathematics
PO Box 513, Eindhoven
The Netherlands.

Some extensions of AUTOMATH: The AUT-4 family.

1. In AUTOMATH (see [1],[2]) we have expressions of degree 1,2,3 and a typing operation that attaches to each expression of degree 2 an expression of degree 1, and to each expression of degree 3 an expression of degree 2. If the typing operation takes P into Q we shall write this here as P E Q. The lines in AUTOMATH are all of the form
<context indicator> , <identifier> , <definitional part> , <category part>, where the category part D has degree 1 or 2 and where the definitional part C (if it is not the empty block opener symbol or the symbol PN) satisfies C E D.

The substitutional mechanism and the abbreviation system are independent of the degrees of the expressions involved. The degrees do make a difference, however, in the rules that express the right to carry out abstraction and application. We shall not recapitulate these rules of AUTOMATH here.

2. The extensions to be considered in this note all have the first of the following two features, and may or may not have the second one.

(i) Alongside with expressions of degrees 1,2,3 we admit expressions of degree 4, and we consider formulas P E Q where P has degree 4 and Q has degree 3. Accordingly, we admit empty blockopener lines and PN-lines with category part of degree 3.

(ii) Definitional equivalence $\stackrel{D}{=}$ is extended as follows: For expressions of degree 4 we take the rules that directly correspond with the rules we have in AUTOMATH for expressions of degrees 1,2,3, but we take the following rule in addition: if $P_1, P_2$ are expressions of degree 4, if $P_1$ E $Q_1$, $P_2$ E $Q_2$, where $Q_1$ and $Q_2$ are expressions of degree 3 satisfying $Q_1 \stackrel{D}{=} Q_2$, then we have $P_1 \stackrel{D}{=} P_2$.

We shall refer to this feature as fourth degree identification.

The AUTOMATH-like languages for which (i) is required, will be called AUT.4-languages.

3. In order to show the ideas behind AUT 4, we first denote some attention to the interpretation of texts in languages of the AUTOMATH family. We necessarily have to be vague about this, since "interpretation" will mean the system of relations between an AUTOMATH book and the "mathematical world". This mathmatical world is not a real world, but the imaginary world of mathematics that has developed in the mind of mathematicians. These mathematicians have been able to discuss that world in natural language, hardly ever getting into serious permanent disagreement, and therefore they feel very confident about it. Nevertheless, it is a strange patchwork of words, formulas and conventions, certainly not easy to describe.

In the mathematical world we say things like "7 is a natural number", " 2 + 2 = 3 is a proposition", and if T is some piece of text we can say "T is a proof for 2 + 2 = 3". Let us indicate the particular use of the word "is" in these sentences by means of the symbol

$$7 \ \varepsilon \ \text{class of all natural numbers}, \tag{3.1}$$

$$2 + 2 = 3 \ \varepsilon \ \text{class of all propositions}, \tag{3.2}$$

$$T \ \varepsilon \ \text{class of all proofs for } 2 + 2 = 4. \tag{3.3}$$

Let us write N and $\pi$ for the classes in (3.1) and (3.2), and let us simply omit the words "class of all proofs" in (3.3). So we get

$$7 \ \varepsilon \ N \ , \tag{3.4}$$

$$2 + 2 = 3 \ \varepsilon \ \pi \ , \tag{3.5}$$

$$T \ \varepsilon \ 2 + 2 = 4. \tag{3.6}$$

In previous reports on AUTOMATH we have recommended the following system of interpretations: the 3, N, etc. of the mathematical world correspond to AUTOMATH expressions that we shall abbreviate here by "3", "N", etc. Now "N", "2+2 = 3", "2+2 = 4" have degree 2, "7" and "T" have degree 3. There is an extra symbol type of degree 1, and we write

"7" E "N" E type, (3.7)

"2+2 = 3" E type, (3.8)

"T" E "2+2 = 4" E type. (3.9)

That is, the symbol type can be interpreted as $\pi$ if we feel like it. This system has both advantages and disadvantages.

An advantage is that we can reduce the number of primitive notions of a book, since there are primitive notions that serve the needs for classes as well as for propositions (for example, the cartesian product of two classes can be specialized to the conjunction of two propositions). But this is also a disadvantage: there can be axioms we want to hold for all propositions and not for all classes. In particular we may wish to phrase the axiom of the excluded third without being forced to accept its equivalent for classes (i.e. Hilbert's universal operator that selects an element from every non-empty set).

This disadvantage can be overcome if we introduce in the AUTOMATH text a primitive "bool" of degree 2, and a function "TRUE" that attaches to every b with b E bool a value TRUE(b) of degree 2 (see [1],[2]). In the interpretation the TRUE(b) corresponds to a proposition, and the things which are E TRUE(b) corresponds to proofs of that proposition. A minor disadvantage is that we have to pass from b's to TRUE(b)'s all the time.

There is also the matter of "type reduction", which we shall briefly discuss presently. Let $\Sigma$ be an expression of degree 2 and assume that for all x with x ε Σ we have derived A(x) E type. Then we may infer in AUT-QE (see [1], section 12.7, [2] p.54) that [x,Σ]A(x) E [x,Σ]type, and it is optional to replace this [x,Σ]type by type (whence [x,Σ]A(x) E type). This is called type reduction, it violates the AUTOMATH law that in A E B the B the B is uniquely determined by A, up to definitional equivalence. In AUTOMATH type reduction is compulsory.

So the feature of AUT-QE is that type reduction can be left undone; at the same time it opens the possibility to start with "let u be a thing with u E [x,Σ]type", which makes it possible to express something with the interpretation "let u be a predicate on Σ".

Experiences with writing AUT-QE seem to have pointed out that type reduction is nice for the cases of class interpretation, and that the possibility to leave type reduction undone is attractive for the cases with propositional interpretation. We might say that we wish type reduction to be effective for classes and not for propositions.

In AUT.4 we have a possibility of an interpretational system that seems to be definitely better than the system described above. Referring to the examples (3.4), (3.5), (3.6), we let "T" have degree 4, "2+2 = 4", "2+2 = 3", "7" get degree 3, "N" and "π" get degree "2", and we admit only a single expression of degree 1, viz. type.

So instead of (3.7) , (3.8) , (3.9) we get

"7" E "N" E type , (3.10)

"2+2 = 3" E "π" E type , (3.11)

"T" E "2+2 = 4" E "π" E type. (3.12)

Propositions and classes now show a difference on the syntactic level: they get different degrees. If we require type reduction, it works for classes and not for propositions.

There is a second syntactic difference with the old system: proofs now get degree 4, and are thus syntactically distinct from things like numbers. The fourth degree identification as described in section 2 (ii) seems to be quite attractive for the case of proofs; it has no consequences for objects like numbers, where the corresponding identification would be utterly unacceptable.

The interpretation of fourth degree identification is what we call <u>irrelevance</u> <u>of</u> <u>proofs</u>. The idea is connected with the general idea of proofs in classical mathematics. In the use of AUTOMATH as described in previous reports ([1],[2]), objects may depend on proofs. For example, the logarithm of a real number is defined for positive numbers only. So actually the log is a function of <u>two</u> variables; and if we use the expression log (p,q), we have to check that p is a real number and q is a proof for the proposition "$p > 0$". If $q_1$ and $q_2$ are different proofs for $p > 0$, the expressions log $(p,q_1)$ and log $(p,q_2)$ are not definitionally equivalent in AUTOMATH. Yet the classical mathematician wants them to be equal (though not necessarely definitionally equal). It causes quite some trouble to achive this, and at every instant where such a thing appears we have to appeal to a place in the book where this kind of equality is expressed. And the text checking computer has to do quite a lot of work, too. Yet it seems so simple: if we are not interested in the difference between $q_1$ and $q_2$ , then we just don't look at them, look only at what they prove. This is what fourth degree identification in AUT.4 can do for us.

4. Let us inspect the various possibilities for language definitions in the AUT.4 family.

(i) There is the possibility to admit <u>quasi-expressions</u> (things like [x,A]<u>type</u>) as expression of degree 1, and to admit or to forbid type reduction.

In connection with what we said in section 3, it seems not necessary to use quasi-expressions. In order to simplify the subsequent discussion we shall restrict ourselves to the case without quasi-expressions. That is, <u>type</u> is only expression of degree 1, and type reduction is compulsory.

(ii) Various things are possible with <u>abstraction</u>. Let A be an expression of degree i (i = 1,2,3) and assume that we have in our book with the context x E A that B(x) E C(x), where B(x) and C(x) have degree j+1 and j, respectively (j = 1,2,3). Then we wish to be allowed to write outside the context x E A, that

$$[x,A]\ B(x) \quad E \quad [x,A]\ C(x) \tag{4.1}$$

if j > 1, and

$$[x,A]\ B(x) \quad E \quad \underline{type} \tag{4.2}$$

if j = 1 (whence C(x) = <u>type</u>). Let us call this (i,j)-abstraction and refer to it as "<u>abstraction from A</u>".

We can choose for which pairs (i,j) we shall admit this in the language definition. It seems to be reasonaible to admit all pairs with i ≠ 1. (One might hesitate about i = 3, but if we do not admit i = 3 here, the passage from the interpretational system (3.4)-(3.6) to the system (3.10)-(3.12) would have a serious drawback.

Entirely independent of the question whether (4.1) and (4.2) are admitted, we agree that the "abstractive expression [x,A]B(x) has the same degree as B(x) (and [x,A] C(x) has the same degree as C(x)).

(iii) Let, in a certain context, q and f be expressions of degree i+1 and j+1, and assume that

$$q \ E\ A\ ,\ f\ E\ [x,A]C(x) \tag{4.3}$$

where A has degree i and C(x) has degree j. Then we can wish to be allowed to write

$$\{q\}f \quad E \quad C(q) \tag{4.4}$$

(which is called (i,j)-application, in this particular case <u>application of</u> f <u>to</u> q). We can choose for which pairs (i,j) ($1 \le i \le 3$, $1 \le j \le 3$) we shall admit this in the language definition. It seems reasonable to admit it for all pairs with $2 \le i \le 3$, $2 \le j \le 3$.

Entirely independent of the question whether (4.4) is admitted, we agree the "applicative expression" {q}f has the same degree as f.

(iv) If j = 3 there is a rule slightly stronger than (iii): if q and f have degree i+1 and 4, respectively, and if

$$q \; E \; A \;, \quad f \; E \; h \;, \quad h \; E \; [x,A]D(x)$$

then

$$\{q\}f \quad E \quad \{q\}h.$$

This rule probably has not much use. It is certainly superfluous if we have both β and η-reduction, for then we can write h as [x,A]{x}h. But even if it is not superfluous it is questionable whether we shall ever need it badly.

(v) Let in a certain context q have degree i+1, and let [x,A]B(x) be an expression of degree j+1 where A has degree i. Then we wish to reduce

$$\{q\}\,[x,A]\,B(x) \quad \text{to} \quad B(q),$$

which is called β-reduction. We have to state for which pairs (i,j) this will be taken as definitional equivalence. It seems reasonable to admit all i and j with $2 \le i \le 3$, $2 \le j \le 4$.

Instead of this ordinary $\beta$-reduction, we can take type-restricted $\beta$-reduction. In that case, the above reduction is allowed only if we can show that $q \in A$.

(vi) If A and B have degrees i and j, we wish to reduce

$$[x,A]\{x\}B \quad \text{to} \quad B$$

($\eta$ -reduction). It is reasonable to admit it for $2 \leq i \leq 3$, $2 \leq j \leq 4$.

(vii) In the interpretations we discussed in section 3 we had use for expressions p of degree 4 only if $p \in A \in$ prop. Let us refer to all other expressions of degree 4 as extras.

One might think it better to ban all extras as long as no interpretation has been agreed upon. On the other hand one may hope to be able to show that that extras do no harm, in the following sense:

Let B be a correct book, and let $\ell$ be a line we wish to add to B. Let $B^*$ be a book with exactly the same PN's as B has, and no others. Assume that $B^*$ contains the line $\ell$, and that neither B nor $\ell$ contain extras. Then there is a correct extension $B^{**}$ of B such that $B^{**} \setminus B$ contains $\ell$ but no PN's and no extras. If this is true, we may say that the language that allows extras is conservative over the one that does not.

Nevertheless there can be some use for extras. Assume we are interested in constructibility of real numbers. Let C be a construction for the real number r, and if "C" and "r" are the expressions corresponding to C and r, we want to have "C" $\in$ "r". By means of axioms we can describe primitive constructions, and primitive ways to obtain new constructions from constructions already known. In that way we can get a theory of constructions in our book. If our language has fourth degree identification then we have construction irrelevance: if something depends on a construction it only depends on the constructed object and on the fact that a construction exists.

References.

[1] N.G. de Bruijn,"The mathematical language AUTOMATH, its usage, and some of its extensions", Symposium on Automatic Demonstration (Versailles December 1968), Lecture Notes in Mathematics, Vol. 125, Springer Verlag, 1970, pp. 29-61.

[2] N.G. de Bruijn, "AUTOMATH, a language for mathematics", Notes (prepared by B.Fawcett) of a series of lectures in the Séminaire de Mathématiques Supérieures, Université de Montreal, 1971.