2274.2

a student prepared the [illegible] c 1970

Description of the invention called:

"Digital Calculating Machine with Fixed Prearranged Program, with Limited Algebraic Keyboard able to Compose Formulas through the Combination of Single Symbolic Elements."

****

To illustrate the invention we state in advance some brief notions on the way in which programs are executed and prepared in a digital computing machine actually known. We consider as an example a three-address machine, however this is done not to limit the range of the invention (which can be applied to machines with 1, 2, 3, or 4 addresses) but only to facilitate the comprehension.

The following premises, as well as the specific description of the invention, make reference to the illustrations, where:

--fig. 1 schematizes the codifying device and zones of the keyboard of the known machine;

--figg. 2 and 3 illustrate two phases during the execution of a program in the known machine;

--fig. 4 schematizes the codifying device with limited algebraic keyboard according to my invention;

—figg. 5 and 6 illustrate the two phases in the automatic computation of a program in the invented machine;
—figg. 7 and 8 illustrate two phases in the execution of a program in the invented machine; and,
—fig. 9 shows the limited algebraic keyboard in the invented machine.

Such a machine consists of: a reader L (see fig. 2) capable of reading paper or magnetic tapes; an internal memory constituted, for example, of 1000 cells, each of which is characterized by a number called its address; furthermore, each cell can contain a decimal number of at most fourteen digits. The contents of a cell can have two meanings: 1) a number; or, 2) an instruction transformed into a number, in other words, a "compiled" instruction. In the chosen example, the compilation is realized as follows:

a) To each ~~operation~~ arithmetic operation, there corresponds a number k according to the following table:

operation symbol
number k

b) Each instruction is a decimal number (of 14 digits) with a fixed structure.

1^a	2^a	3^a	4^a	5^a	6^a	7^a	8^a	9^a	10^a	11^a	12^a	13^a	14^a	dec. digits
0	1	3	2	0	4	0	1	3	3	0	9	2	1	~~dec. digits~~ compiled inst.
	addr.of 1st oper.				op.		addr. of 2nd oper.				addr. of result			

In the above example, the instruction corresponding to the number n = 01320401330921 is interpreted as follows: "The number contained in the address 132 (1st operand of the operation) is divided (since k = 4) by the number contained in the address 133 (2nd operand); the result of the operation is stored in the address 921." This work of interpreting is done by the control P (fig. 3) which executes the instruction and fetches from the memory the next instruction. The numbers are transferred for this purpose into the arithmetic unit U.A. where the operation described by the instruction is executed. Furthermore there are compiled instructions which transfer the result to a teletypewriter T which prints on a sheet of paper F (fig. 3) or to a unit R which records on magnetic or paper tape.

Similarly, there are compiled instructions which read a number from an input tape through 1 and transfer it to a certain set address. The structure of the three-address machine and a compiled instruction have been described in general terms. We consider now briefly how this machine works when it executes a program, as illustrated in figures 2 and 3. First, (fig. 2) a variable program (p.vp.) (which is called variable to distinguish it from the fixed internal program characteristic of the present invention, to be discussed later) is transferred to the internal memory of the machine (the dark lines indicate in each figure the specific part of the machine functioning during that phase). Then (fig. 3) the numerical data relative to the program, which are recorded on the input tape by a device which it is not necessary to describe, are communicated to the machine which it is executing the program, and finally the results are transferred the the teletypewriter T which prints them on F.

The preparation of the compiled instructions for a specific program is performed in advance ~~through a~~ by a suitable device illustrated schematically in fig. 1. It consists of:

a) a keyboard T --
b) an electromechanical device C connected to it --
c) a magnetic or paper tape on which the device C records the series of compiled instructions.

The keyboard consists of 4 zones T_1, T_2, T_3, and T_4. In the first, third, and forth zones there are keys with letters (a,b,c,d, . . .). In the second zone, the keys indicate the operations (e. g. +, -, . , :, etc.). In the 1st, 3rd, and 4th zones, all the letters are present (in each of these 3 zones there are, for example, 30 or more keys). This partitioning into 4 zones is due to the fact that each zone has a particular significance. With reference to the decimal structure described before, the ~~present~~ pressing of a key in the first zone generates the first address in the instruction, the pressing of a key in the second zone generates the number k corresponding to the operation, and so on.

If we want to program for example the computation of the sum

$$(1) \quad a + b = x ,$$

we must press the key "a" in the first zone, the key "+" in the second, the key "b" in the third, and the key "x" in the forth. If this procedure is not followed, for example, if we press more than one key per zone, we generate an instudtion without meaning.

It follows that to each key in the 1st, 3rd, and 4th zones there corresponds an address, so that only 26 different addresses can be used with this method. A remedy to this inconvenience is to add 3 other zones T_{I}, T_{III}, and T_{IV} in the keyboard under the 1st, 3rd, and 4th zones, containing numeric keys numbered, for example, from 1 to 30 in each zone. With the convention of pressing always after a lettered key a numeric key in the zone below, it is as if the keyboard were 30 times larger, or as if we were using 30 different alphabets.

The capacity of the keyboard is extended to 880 different addressed corresponding to each letter-number couple. Nevertheless, however, this extension of the keyboard (about 200 keys in the example cited above), which makes the machine move complex and less comfortable, does not permit the compilation of operations more complex than those of the binary type:

$$V_1 \text{ op } V_2 = V_3 ,$$

where V_1, V_2, and V_3 are generic addresses and op is for example, one of the operations +, -, ., :, etc.

If the problem to be compiled allows chains of operations (corresponding to formulas with more than one operation and more that two operands and eventually with parentheses), they must be separated by the operator into a succession of binary operations, which makes more tiresome the programming and more probably the mistakes.

In the invented machine, great simplifications have been obtained in the preparation of programs, which is manifested especially in the simplification of the compiling device, and in the substitution of the old keyboard with a new keyboard with a simplified structure, and extended capabilities.

The new algebraic keyboard schematized in fig. 9 consists of about 40 keys, not divided into zones, corresponding to letters, symbols, and operations.

We must mention expressly the keys (,) , $\rightarrow$, and $\downarrow$, which don't appear in the pressent machines. With the new keyboard, we have the following advantages:

1) The number of keys is reduced by a factor of five.

2) The keys have ne positional value which simplifies the external appearance of the keyboard as well as the connected electromechanical device. The instruction (1) is recorded by pressing successively the keys

a , +, b, $\rightarrow$, and c.

We observe that the key $\rightarrow$ replaces the equal sign and has the basic function of separating the result from the other symbols in the formula.

3)

3) Not only is the keyboard simpler but more efficient because it is "algebraic". In fact a chain of operations with a nearly unlimited number of operands and operators can be programmed. For example, an expression of the type

$$a+b.c-d:f+g.h.k:m \rightarrow x$$

can be used as a program, and recorded by pressing the corresponding keys successively as we do with a common typewriter.

4) It must be emphasized that the use of parentheses is exactly analogous to their use in common algebraic expressions as in the following example:

$$(((a+b).c):d) \rightarrow y$$

can also be programmed. We note futhermore that the parentheses of every order can be represented by by the same two symbols, since their position determines the degree of inclusion assumed.

5) We can extend the power of the keyboard through the use of the key $\Downarrow$ which should be pressed immediately before the lettered keys if it is needed.

This key makes possible, without the introduction of numerical keys or extra zones as in the present machines, the programming of operations which access numbers which are in addresses different from the 26 addresses corresponding to the letters of the alphabet.

Furthermore the use of this key allows us to write once and for all a formula that must be computed many times successively always with different values of data and results. This is explained with an example.

Let's suppose we have recorded the following formula by pressing the keys

$$\downarrow a \;:\; \downarrow b \rightarrow \downarrow c$$

and let's suppose that the compiled instruction is represented by the number n' where

$n' = 1\ 0\ 0\ 1\ 0\ 4\ 1\ 0\ 0\ 2\ 1\ 0\ 0\ 3$, and a corres. to addr. 1
b corres. to addr. 2
c " " " 3,

while to the instruction

$$a : b \rightarrow c$$

corresponds the number m where

$$m = 0\ 0\ 0\ 1\ 0\ 4\ 0\ 0\ 0\ 2\ 0\ 0\ 0\ 3\ .$$

As can be seen, the presence of 3 units in the columns 1, 7, and 11, respectively, in the number n' denote the presence of three keys. This presence assigns a different meaning to the intruction which is interpreted by the control P as follows: "The number contained in the address which in turn is contained in the address a (or 1), is divided by the number in the address which in turn is contained in the address b (or 2); the result of the operation is transferred to the address pointed to by c (or 3).

If at the moment of execution of this instruction, the contents of the address a were 132, those of the address b were 133, and those of the address c were 921, then the instruction n' would be equivalent to the instruction n on page 1. If the contents of the addresses a, b, and c were the numbers 1, 2, and 3, respectively, then the instruction n' would be equivalent to the instruction m.

The key $\downarrow$ allows (always supposing the machine is able to execute the corresponding operation) the programming of operations with numbers which are contained in any address, in particular, in successive addresses indicated at successive times.

The following description has the purpose of illustrating how we can obtain the above advantages.

By pressing each key on the limited algebraic keyboard (T' in fig. 4 and fig. 9) we obtain the recording of a number on magnetic or paper tape using the device C' in fig. 4, so that we obtain a series of numbers called the specific algebraic program (p.a.p.).

These numbers correspond to the algebraic formulas 2), 3), and 4) on pages 8 and 9.

In the machine we make use of a fixed program, called internal, which makes it possible for the machine to operate on the specific algebraic program recorded on the tape as indicated before.

Technically, two types of realizations are possible for the fixed program:

A) A recording on magnetic or paper tape as indicated in figure 5;

B)

B) Incorporation of it in the machine using a ~~pmem~~ special device consisting of circuits, [redacted] control elements (relays, tubes) which simulate the program.

As an example, we describe a realization of type A). Furthermore, we should notice that the use of the machine to elaborate programs is a novelty by itself. The phases of the work are the following:

1) The fixed program (p.f.) is tranferred to the machine and recorded in the memory (fig. 5). This phase is executed only once.
2) The specific algebraic program (p.a.p.) is recorded using on the tape using the device T'C' (fig. 4).
3) The tape containing the p.a.p. is placed at the input to the machine (fig. 6) and follows a phase of computation similar to that described in fig. 3, where is place of the ~~nmmhem~~ numerical data there is the p.a.p. and in place of the specific program is the fixed program. As a result of this computation the specific program ~~inxmemoryxandxonxthexoutput~~ (p.v.p.) ~~inxmemoryxandxonxthexoutputxtape~~ relative to the p.a.p. is recorded on the output tape.

4) The p.v.p. is transferred into the internal memory of the machine is a different place than that occupied by the fixed program (p.f.) (fig. 7).
5) A phase of computation (fig. 8) identical to that described in fig. 3 then follows.

In the realization of case B, the fixed program is incorporated in the machine; phase 1 disappears, phases 2 through 5 are analogous to case A, with the difference that place occupied by the p.f. in the memory is free.

Summarizing, the internal program has the function of decomposing the specfic program (variable each step) in single elementary operations because all the machines (and consequently, the present machine) can't execute more one operation at a time.

While in the old machines we cannot introduce programs with chains of operations, with the use of the fixed internal program we can introduce programs consisting of chains of operations with unlimited numbers of terms and operators.

The use of the fixed internal program, once introduced into the machine, allows the simplifications and perfections described, obtaining a greater efficiency in the machine.

To make the concept clear, an algebraic illustration of the fixed program follows.

We describe the operation of the machine from the beginning of phase 3. For this purpose, we use a method of representation similar to that of Von Neumann (structural diagrams) illustrated in figure 10. In the diagram we have small circles containing capital letters and directed edges designated by small letters. Each letter in circle represents a particular group of operations; each directed edge indicate the order of execution between the groups of operations it connects. If from a letter diverge several edges to different letters, this means that the group of operations represented by the initial letter is followed by one of the groups represented by the final letters. Which group is choosen to the exclusion of the others depends on the structure of the p.a.p. (pressed before on the keyboard). We observe for example: A: this group of operations can be follows by A' or B, depending on the cases as we said before.

In the following, each capital letter is followed by a description of the group of operations it represents; each small letter, by the condition which determines the choice of that particular edge to the exclusion of the others (e. g. : "The number of right parentheses does not exceed the number of left parentheses" or "the penultimate symbol is an operation and the last symbol is a variable", etc.) If there is only one edge, then there is no small letter since there is no choice. To simplify matters, we assign to each edge the small letter corresponding to the capital letter in the circle the edge points to.

The beginning of the fixed program is at A. The group Ω, which appears twice in the diagram, is characterized by the lack of edges leaving from it.

Ω is the command to stop. When an incorrect formula is written on the keyboard, then the condition which takes us to Ω arises and the machine stops.

The letters in the graph of figure 19 mean:

Ω = Stop

A = The first compiled symbol written on the tape is read by the machine.

- ω The first symbol is either a right parenthesis ~~and~~ or $\rightarrow$ or ~~xxxxxxx~~ an operation symbol (see fig. 9, first row).
- a' The first symbol is a variable (see fig. 9: a letter or a letter preceded by $\downarrow$).
- b The first symbol is a ~~right parenthesis~~ left parenthesis.

B = The succeeding compiled symbol is read by the machine.

- b_1 The symbol is not a parenthesis.
- b_2 The symbol is a left parenthesis.
- b_3 The symbol is a right parenthesis.

B_1 = The last two symbols are read into the machine determine the rest of the operations.
Near to each small letter are indicated the two symbols which determine the operation
(op and V are abbreviations of operations and variables).

b_4 :))   f : (V   c : V)   g : $\rightarrow$V   b: ((

d : op (h : op V   j : V op   e :)$\rightarrow$ i :) op

ω : All combinations of two symbols not indicated above.

I = Determination of the first six letters in the compiled instruction.

ω The number of ~~left~~ right parentheses exceeds the number of left parentheses.

b The number of right parentheses does not exceed the number of left parentheses.

B_4 = Check that the number of right parentheses read in by the machine does not exceed the number of left parentheses.

ω As in I.

b As in I.

D = Completion of the compiled instruction, that is, the determination of the 7th through the 24th digits of the instruction ~~and~~ (see page 3), and its provisional transfer to a precalculated address in the memory, on the basis of the degree of inclusion of the last parenthesis read in. See B for what then follows.

F = Determination of the first 4 digits of the compiled instruction. Then, see B.

H = Determination of the second operand of the instruction (7--10). Then, see B.

J = Determination of the operation (6th digit) of the compiled instruction. Then, see B.

G = Completion of the last compiled instruction of a formula (determination of digits ~~11~~--14) and ~~printing of the~~ output of the instruction on the tape of the p.v.p. Then, see A.

E = Determination of the second operand of the compiled instruction (the last of a formula);

- ω The number of left parentheses is different from the number of right parentheses.
- e_1 The number of right parentheses is equal to the number of left parentheses.

E_1 = Transfer of the compiled instruction from the prearranged address to the tape of the p.v.p. so that the right sequence of instruction on the tape is respected.

- e_1 Not all the compiled instruction for a given formula have been transferred to the tape of the p.v.p.
- b All the compiled instructions (with the exception of the last one) have been transferred to the tape of the p.v.p.

B_2= Computation of the degree of inclusion of the last left parenthesis introduced into the machine. Then, see B_1.

B_3= Computation of the degree of inclusion of the last right parenthesis introduced into the machine. Then, see B_1.

A'= The second compiled symbol is introduced into the machine.

ω The second symbol is a parenthesis or an operation different from +, −, ., :,   , or is a variable.

a''' The second symbol is   .

a'^{V} The second symbol is + or −.

a^{V} The second symbol is . or :.

A'''=The third compiled symbol is introduced into the machine. A compiled instruction corresponding to the three symbols read in is formed and outputed on the tape of the p.v.p.

ω The third symbol is not a variable.

a The third symbol is a variable.

C = The same as in D. Then, see B.

A'^{V}=The first compiled instruction of formed and put out on the tape of the p.v.p.

A^{V} = (Same as A'^{V}.)

A'' = A new compiled symbol is read in.
   ω The new symbol is not a variable;
   f' The penultimate symbol read in is ;
   a''_{bis} The penultimate symbol is not a .

F' = The last compiled instruction of a formula is written out on the tape of the p.v.p. Then, see A.

A''_{bis} = Another symbol is read in.
   ω The symbol is not one of the following: +, −, ., :, ;
   b' The penultimate operation symbol read in was . or : ;
   e' The penultimate operation symbol read in was + or − and the last was . or : ;
   d' The penultimate operation symbol read in was + or − and the last was + or − or .

B' = A new compiled instruction is formed and written out on the tape of the p.v.p.

a" The last operation symbol read in was . or : ;
c' The last operation symbol read in was not either . or : (hence, is +, -, or).

C'= A new compiled instruction is formed and written out on the tape of the p.v.p. Then, see A".

D'= The same. Then, see A".

E'= The same. Then see A".

Example -- The execution of the fixed program of the formula mentioned in 3) (page 9) is equivalent to the execution of the following group of operations in sequence:

A A' A'^{V} A" A''_{bis} E' A" A''_{bis} B' C' A" A''_{bis}

E' A" A''_{bis} B' C' A" A''_{bis} E' A" A''_{bis} B' A" A''_{bis} B'

A" A''_{bis} B' C' A" F'

Instead, for the formula mentioned in 4) (page 9), the corresponding sequence of groups of operations is the following:

A B B_2 B_1 B B_2 B_1 F B B_1 J B B_1 H B B_3 B_1 C B B_1 I B+B_1 H

B B_3 B_1 C B B_1 I B B_1 H B B_3 B_1 C B B_1 E E_1 E_1 E_1 B B_1 G

In the first of the two examples, we obtain on the tape of the p.v.p. a sequence of compiled instructions equivalent to the following sequence of binary operations.

$$a \longrightarrow S$$

$$b \longrightarrow X$$

$$X,c \longrightarrow X$$

$$S+X \longrightarrow S$$

$$d \longrightarrow S$$

$$X:f \longrightarrow X$$

$$S-X \longrightarrow S$$

$$g \longrightarrow X$$

$$X.h \longrightarrow X$$

$$X.l \longrightarrow X$$

$$X:m \longrightarrow X$$

$$S+X \longrightarrow S$$

$$S \longrightarrow x$$

In the second of the ~~tw~~ two examples, we obtain instead on the tape of the p.v.p.

$$a + b \longrightarrow x_1$$

$$x_1 . c \longrightarrow x_2$$

$$x_2 : d \longrightarrow x_3$$

$$x_3 \longrightarrow y$$

While it should be clear to the mathematician from the previous remarks how the program is set up, for the novice we add some extra remarks.

We consider all possible algebraic formulas that can be introduced into the machine by operating the keyboard. We observe which form they should have in order to be received as compiled instructions.

Observing all the possible starting points and the necessary final points, we construct a fixed program in order to prearrange the right response of the machine so that the keyboard and the fixed program are conditioned recipricaully.

Summary

1) We claim the invention of a digital calculating machine and program, characterized by a fixed prearranged program for the elaboration of variable programs (where variable means that they may change from time to time) representing algebraic programs of unlimited length.

2) We claim the invention of a machine as in 1), in which the fixed prearranged program is recorded on paper or magnetic tape or incorporated in some device inside the machine.

3) The machine as in 1) and 2) is characterized furtherby a limited algebraic keyboard (without zones) in which each key produces a number without positional value according to a given system with possibilities of composing algebraic formulas of unlimited length with an unlimited number of parentheses.

4) The machine as in 3) has on its keyboard the additional keys $\rightarrow$ and $\downarrow$; the latter allows the programming of operations with numbers contained at any address, and furthermore, allows us to change the addresses at successive times, which is particularly useful if we must repeat the same operation or groups of operations with different sets of data.

2274.2

a student prepared this for me c 1970

Description of the invention called:

"Digital Calculating Machine with Fixed Prearranged Program, with Limited Algebraic Keyboard able to Compose Formulas through the Combination of Single Symbolic Elements."

****

To illustrate the invention we state in advance some brief notions on the way in which programs are executed and prepared in a digital computing machine actually known. We consider as an example a three-address machine, however this is done not to limit the range of the invention (which can be applied to machines with 1, 2, 3, or 4 addresses) but only to facilitate the comprehension.

The following premises, as well as the specific description of the invention, make reference to the illustrations, where:

--fig. 1 schematizes the codifying device and zones of the keyboard of the known machine;

--figg. 2 and 3 illustrate two phases during the execution of a program in the known machine;

--fig. 4 schematizes the codifying device with limited algebraic keyboard according to my invention;