Lecture to the London Mathematieal Society on 20 February 1947 [*]

A. M. Turing

The automatic computing engine now being designed at N.P.L. is a typical large scale electronic digital computing machine. In a single lecture it will not be possible to give much technical detail of this machine, and most of what I shall say will apply equally to any other machine of this type now being planned.

From the point of view of the mathematician the property of being digital should be of greater interest than that of being electronic. That it is electronic is certainly important because these machines owe their high speed to this, and without the speed it is doubtfül if financial support for their construction would be forthcoming. But this is virtually all that there is to be said on that subject. That the machine is digital however has more subtle significance. It means firstly that numbers are represented by sequences of digits which can be as long as one wishes. One can therefore work to any desired degree of accuracy. This accuracy is not obtained by more careful machining of parts, control of temperature variations, and such means, but by a slight increase in the amount of equipment in the machine. To double the number of significant figures used would involve increasing the equipment by a factor definitely less than two, and would also have some effect in increasing the time taken over each job. This is in sharp contrast with analogue machines, and continuous variable machines such as the differential analyser, where each additional decimal digit required necessitates a complete redesign of the machine, and an increase in the cost by perhaps as much as a factor of 10. A second advantage of digital computing machines is that they are not restricted in their applications to any particular type of problem. The differential analyser is by far the most general type of analogue machine yet produced, but even it is comparatively limited in its scope. It can be made to deal with almost any kind of ordinary differential equation, but it is hardly able to deal with partial differential equations at all, and certainly cannot manage large numbers of linear simultaneous equations, or the zeros of polynomials. With digital machines however it is almost literally true that they are able to tackle any computing problem. A good working rule is that the ACE can be made to do any job that could be done by a human computer, and will do it in one ten-thousandth of the time. This time estimate is fairly reliable, except in cases where the job is too trivial to be worth while giving to the ACE.

Some years ago I was researching on what might now be described as an investigation of the theoretical possibilities and limitations of digital computing machines. I considered a type of machine which had a central mechanism, and an infinite memory which was contained on an infinite tape. This type of machine appeared to be sufficiently general. One of my conclusions was that the idea of a 'rule of thumb' process and a 'machine process' were synonymous. The expression 'machine process' of course means one which could be carried out by the type of machine I was considering. lt was essential in these theoretical arguments that the meinory should be infinite. It can easily be shown that otherwise the machine can only execute periodic operations. Machines such as the ACE may be regarded as practical versions of this same type of machine. There is at least a very close

---

* A.M. Turings ACE Report of 1946 and other papers – Vol 10 "In the Charles Babbages Repring Series for the History of Computing", (B.E. Carpenter, B.W. Doran, eds.) The MIT Press, 1986

analogy. Digital computing machines all have a central mechanism or control and some very extensive form of memory. The memory does not have to be infinite, but it certainly needs to be very large. In general the arrangement of the memory on an infinite tape is unsatisfactory in a practical machine, because of the large amount of time which is liable to be spent in shifting up and down the tape to reach the point at which a particular piece of information required at the moment is stored. Thus a problem might easily need a storage of three million entries, and if each entry was equally likely to be the next required the average journey up the tape would be through a million entries, and this would be intolerable. One needs some form of memory with which any required entry can be reached at short notice. This difficulty presumably used to worry the Egyptians when their books were written on papyrus scrolls. It must have been slow work looking up references in them, and the present arrangement of written matter in books which can be opened at any point is greatly to be preferred. We may say that storage on tape and papyrus scrolls is somewhat *inaccessible*. It takes a considerable time to find a given entry. Memory in book form is a good deal better, and is certainly highly suitable when it is to be read by the human eye. We could even imagine a computing machine that was made to work with a memory based on books. It would not be very easy but would be immensely preferable to the single long tape. Let us for the sake of argument suppose that the difficulties involved in using books as memory were overcome, that is to say that mechanical devices for finding the right book and opening it at the right page, etc. etc. had been developed, imitating the use of human hands and eyes. The information contained in the books would still be rather inaccessible because of the time occupied in the mechanical motions. One cannot turn a page over very quickly without tearing it, and if one were to do much transportation, and do it fast, the energy involved would be very great. Thus if we moved one book every millisecond and each was moved ten metres and weighed 200 grams, and if the kinetic energy were wasted each time we should consume 10^{10} watts, about half the country's power consumption. lf we are to have a really fast machine then, we must have our information, or at any rate a part of it, in a more accessible form than can be obtained with books. It seerns that this can only be done at the expense of compactness and economy, e.g. by cutting the pages out of the books, and putting each one in to a separate reading mechanism. Some of the methods of storage which are being developed at the present time are not unlike this.

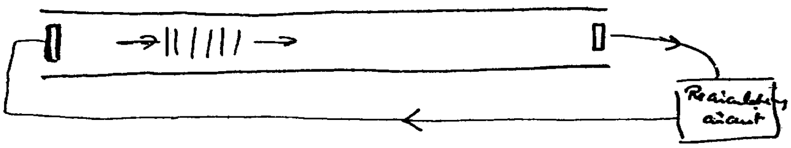
lf one wishes to go to the extreme of accessibility in storage mechanisms one is liable to find that it is gained at the price of an intolerable loss of compactness and economy. For instance the most accessible known form of storage is that provided by the valve flip-flop or Jordan Eccles trigger circuit. This enables us to store one digit, capable of two values, and uses two thermionic valves. To store the content of an ordinary novel by such means would cost many millions of pounds. We clearly need some compromise method of storage which is more accessible than paper, film etc, but more economical -in space and money than the straightforward use of valves. Another desirable feature is that it should be possible to record into the memory from within the computing machine, and this should be possible whether or not the storage already contains something, i.e. the storage should be *erasible*.

There are three main types of storage which have been developed recently and have these properties in greater or less degree. Magnetic wire is very compact, is erasible, can be recorded on from within the machine, and is moderately

seminarTEXT

accessible. There is storage in the form of charge patterns on the screen of a cathode ray tube. This is probably the ultimate solution. It could eventually be nearly as accessible as the Jordan Eccles circuit. A third possibility is provided by acoustic delay lines. They give greater accessibility than the magnetic wire, though less than the C.R.T type. The accessibility is adequate for most purposes. Their chief advantage is that they are already a going concern. It is intended that the main memory of the ACE shall be provided by acoustic delay lines, consisting of mercury tanks.

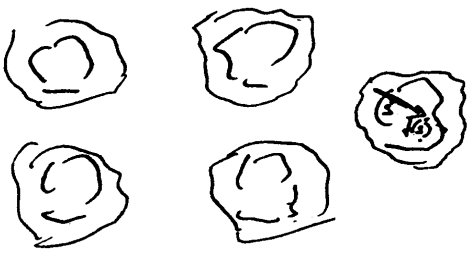
The idea of using acoustic delay lines as memory units is due I believe to Eckert of Philadelphia University, who was the engineer chiefly responsible for the Eniac. The idea is to store the information in the form of compression waves travelling along a column of mercury. Liquids and solids will transmit sound of surprisingly high frequency, and it is quite feasible to put as many as 1000 pulses into a single 5' tube. The signals may be conveyed into the mercury by a piezo-electric crystal, and also detected at the far end by another quartz crystal. A train of pulses or the information



which they represent may be regarded as stored in the mercury whilst it is travelling through it. lf the information is not required when the train emerges it can be fed back into the column again and again until such time as it is required. This requires a 'recirculating circuit' to read the signal as it emerges from the tank and amplify it and feed it in again. lf this were done with a simple amplifier it is clear that the characteristics of both the tank and the amplifier would have to be extremely good to permit the signal to pass through even as many as ten times. Actually the recirculating circuit does something slightly different. What it does may perhaps be best expressed in terms of point set topology. Let the plane of the diagram represent the space of all possible signals. I do not of course wish to imply that this is two dimensional. Let the füntion f be defined for arguments in this signal space and have values in it. In fact let $f(s)$ represent the effect on the signal s when it is passed through the tank and the recirculating mechanism. We assume however that owing to thermal agitation the effect of recirculation may be to give any point within a circle of radius δ of $f(s)$. Then a necessary and sufficient condition that the tank can be used as a storage which will distinguish between N different signals is that there must be N sets $E_1 \ldots E_N$ such that if F_r is the set of points within distance ε of E_r

$$s \in F_r \subset f(s) \in E_r$$

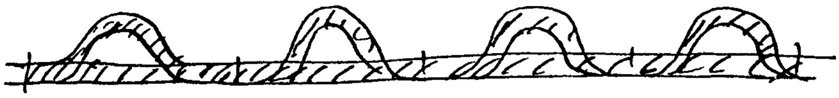
and the sets F_r are disjoint. It is clearly sufficient for we have only then to ensure that the signals initially fed in belong to one or other of the sets F_r and it will remain in the set after any number of recirculations, without any



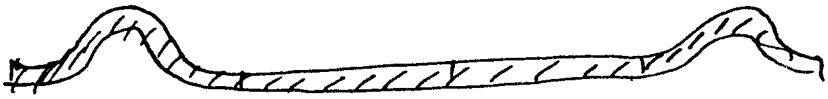
seminarTEXT

danger of confusion. It is necessary for suppose $s_1 \ldots s_N$ are signals which have different meanings and which can be fed into the machine at any time and read out later without fear of confusion.

Let E_r be the set of signals which could be obtained for s, by successive applications of f and shifts of distance not more than ε. Then the sets E_r are disjoint [two lines indecipherable - Ed.]. In the case of a mercury delay line used for N = 16 the set would consist of all continuous signals within the shaded area.



One of the sets would consist of all continuous signals lying in the region below. It would represent the signal 1001.



In order to put such a recirculation system into effect it is essential that a clock signal be supplied to the memory system so that it will be able to distinguish the times when a pulse if any should be present. lt would for instance be natural to supply a timing sine wave as shown above to the recirculator.

The idea of a process f with the properties we have described is a very common one in connection with storage devices. It is known as 'regeneration' of storage. It is always present in some form, but sometimes the regeneration is as it were naturally occuring and no precautions have to be taken. In other cases special precautions have to be taken to improve such an f process or else the impression will fade.

The importance of a clock to the regeneration process in delay lines may be illustrated by an interesting little theorem. Suppose that instead of the condition $s \in Fr \subset f(s) \in E_r$ we impose a stronger one, viz. $f^n(s) \rightarrow c_r$ if $s \in Er$ i.e. there are ideal forms of the distinguishable signals, and each admissible signal converges towards the ideal form after recirculating. Then we can show that unless there is a clock the ideal signals are all constants. For let U_α represent a shift of origin, i.e. $U_\alpha\, s(t) = s(t + \alpha)$. Then since there is no clock the properties of the recirculator are the same at all times andf therefore commutes with U_α. Then $f\, U_\alpha(c_r) = U_\alpha f(c_r) = U_\alpha c_r$ for $f(c_r) = c_r$ since cr is an ideal signal. But this means that $U_\alpha(c_r)$ is an ideal signal, and therefore for sufficiently small α must be c_r since the ideal signals are discrete. Then for any β and sufficiently large u, $ß/u$ will be sufficiently small and $U_{ß/u}(c) = c$. But then by iteration $c = U^u{}_{ß/u}(c) = U_ß(c)$ i.e. $c(t + ß) = c(t)$. This means that the ideal signal c is a constant.

We might say that the clock enables us to introduce a discreteness into time, so that time for some purposes can be regarded as a succession of instants instead of a continuous flow. A digital machine must essentially deal with discrete objects,

seminarTEXT

and in the case of the ACE this is made possible by the use of a clock. All other digital computing machines except for human and other brains that I know of do the same. One can think up ways of avoiding it, but they are very awkward. I should mention that the use of the clock in the ACE is not confined to the recirculation process, but is used in almost every part.

lt may be as well to mention some figures connected with the mercury delay line as we shall use it. We shall use five foot tubes, with an inside diameter of half an inch. Each of these will enable us to store 1024 binary digits. The unit I have used here to describe storage capacity is self explanatory. A storage mechanism has a capacity of m binary digits if it can remember any sequence of m digits each being a 0 or a 1. The storage capacity is also the logarithm to the base 2 of the number of different signals which can be remembered, i.e. log, N. The digits will be placed at a time interval of one microsecond, so that the time taken for the waves to travel down the tube is just over a millisecond. The velocity is about one and a half kilometres per second. The delay in accessibility time or average waiting for a given piece of information is about half a millisecond. In practice this is reduced to an effective 150 ps. The füll storage capacity of the ACE available on Hg delay lines will be about 200,000 binary digits. This is probably comparable with the memory capacity of a minnow.

I have spent a considerable time in this lecture on this question of memory, because I believe that the provision of proper storage is the key to the problem of the digital computer, and certainly if they are to be persuaded to show any sort of genuine intelligence much larger capacities than are yet available must be provided. In my opinion this problem of making a large memory available at reasonably short notice is much more important than that of doing operations such as multiplication at high speed. Speed is necessary if the machine is to work fast enough for the machine to be commercially valuable, but a large storage capacity is necessary if it is to be capable of anything more than rather trivial operations. The storage capacity is therefore the more fundamental requirement.

Let us now return to the analogy of the theoretical computing machines with an infinite tape. It can be shown that a single special machine of that type can be made to do the work of all. lt could in fact be made to work as a model of any other machine. The special machine may be called the universal machine, it works in the following quite simple manner. When we have decided what machine we wish to imitate we punch a description of it on the tape of the universal machine. This description explains what the machine would do in every configuration in which it might find itself. The universal machine has only to keep looking at this description in order to find out what it should do at each stage. Thus the complexity of the machine to be imitated is concentrated in the tape and does not appear in the universal machine proper in any way.

lf we take the properties of the universal machine in combination with the fact that machine processes and rule of thumb processes are synonymous we may say that the universal machine is one which, when supplied with the appropriate instructions, can be made to do any rule of thumb process. This feature ig paralleled in digital computing machines such as the ACE. They are in fact practical versions of the universal machine. There is a certain central pool of electronic equipment, and a large memory. When any particular problem has to be

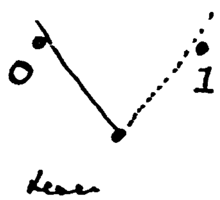
seminarTEXT

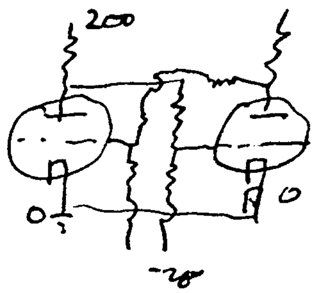
handled the appropriate instructions for the computing process involved are stored in the memory of the ACE and it is then 'set up' for carrying out that process.

I have indicated the main strategic ideas behind digital computing machinery, and will now follow this account up with the very briefest description of the ACE. It may be divided for the sake of argument into the following parts

Memory
Control
Arithmetic part
Input and output

I have already said enough about the memory and will only repeat that in the ACE the memory will consist mainly of 200 mercury delay lines each holding 1024 binary digits. The purpose of the control is to take the right instructions from the memory, see what they mean, and arrange for them to be carried out. It is understood that a certain 'code of instructions' has been laid down, whereby each Vord' or combination of say 32 binary digits describes some particular operation. The circuit of the control is made in accordance with the code, so that the right effect is produced. To a large extent we have also allowed the circuit to determine the code, i.e. we have notjust thought up an imaginary'best code'and then found a circuit to put it into effect, but have often simplified the circuit at the expense of the code. It is also quite difficult to think about the code entirely in abstracto without any kind of circuit. The arithmetic part of the machine is the part concerned with addition, multiplication and any other operations which it seems worth while to do by means of special circuits rather than through the simple facilities provided by the control. The distinction between control and arithmetic part is a rather hazy one, but at any rate it is clear that the machine should at least have an adder and a multiplier, even if they turn out in the end to be part of the control. This is the point at which I should mention that the machine is operated in the binary scale, with two qualifications. Inputs from externally provided data are in decimal, and so are outputs intended for human eyes rather than for later reconsumption by the ACE. This is the first qualification. The second is that, in spite of the intention of binary working there can be no bar on decimal working of a kind, because of the relation of the ACE to the universal machine. Binary working is the most natural thing to do with any large scale computer. It is much easier to work in the scale of two than any other, because it is so easy to produce mechanisms which have two positions of stability: the two positions may then be regarded as representing 0 and 1. Examples are lever as diagram, Jordan Eccles circuit, thyratron. lf one is concerned with a



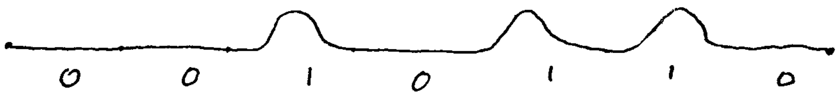


small scale calculating machine then there is at least one serious objection to binary working. For practical use it will be necessary to build a converter to transform numbers from the binary form to the decimal and back. This may well

seminarTEXT

be a larger undertaking than the binary calculator. With the large scale machines this argument carries no weight. In the first place a converter would become a relatively small piece of apparatus, and in the second it would not really be necessary. This last statement sounds quite paradoxical, but it is a simple consequence of the fact that these machines can be made to do any rule of thumb process by remembering suitable instructions. In particular it can be made to do binary decimal conversion. For example in the case of the ACE the provision of the converter involves no more than adding two extra delay lines to the memory. This situation is very typical of what happens with the ACE. There are many fussy little details which have to be taken care of, and which, according to normal engineering practice would require special circuits. We are able to deal with these points without modification of the machine itself, by pure paper work, eventually resulting in feeding in appropriate instructions.

To return to the various parts of the machine. I was saying that it will work in the scale of two. It is not unnatural to use the convention that an electrical pulse shall represent the digit 1 and that absence of a pulse shall represent a digit 0. Thus a sequence of digits 00 10 110 would be represented by a signal like



where the time interval might be one microsecond. Let us now look at what the process of binary addition is like. In ordinary decimal addition we always begin from the right, and the same naturally applies to binary. We have to do this because we cannot tell whether to carry unless we have already dealt with the less significant columns. The same applies with electronic addition, and therefore it is convenient to use the convention that if a sequence of pulses is coming down a line, then the least significant pulse always comes first. This has the unfortunate result that we must either write the least significant digit on the left in our binary numbers or else make time flow from right to left in our diagrams. As the latter alternative would involve writing from right to left as well as adding in that way, we have decided to put the least significant digit on the left. Now let us do a typical addition. Let us write the carry digits above the addends.

Carry		0	1	1	1	1	1	0	0	1	1
A	0	1	1	0	1	1	0	0	1	0	1..
B	0	1	1	1	0	1	0	0	1	1	...
		0	1	0	0	1	1	0	0	0	

Note that I can do the addition only looking at a small part of the data. To do the addition electronically we need to produce a circuit with three inputs and two outputs.

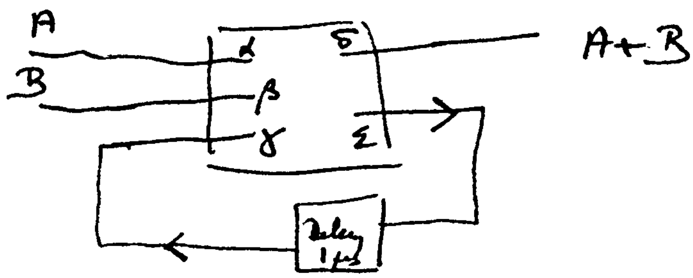
seminarTEXT

Inputs		*Outputs*	
Addend A	α	Sum	δ
Addend B	β	Carry	ε
Carry from last column	γ		

This circuit must be such that

If no. of 1's on inputs α, β, γ is	0	Then sum	0	and	0
	1	δ	1	carry	0
	2	is	0	ε	1
	3		1	is	1

It is very easy to produce a voltage proportional to the number of pulses on the inputs, and one then merely has to provide a circuit which will discriminate between four different levels and put out the appropriate sum and carry digits. I will not attempt to describe such a circuit; it can be quite simple. When we are given the circuit we merely have to connect it up with feedback and it is an adder. Thus:



lt will be seen that we have made use of the fact that the same process is used in addition with each digit, and also the fact that the properties of the electrical circuit are invariant under time shifts, at any rate if these are multiples of the clock period. It might be said that we have made use of the isomorphism between the group of these time shifts and the multiplicative group of real numbers to simplify our apparatus, though I doubt if many other applications of this principle could be found.

It will be seen that with such an adder the addition is broken down into the most elementary steps possible, such as adding one and one. Each of these occupies a microsecond. Our numbers will normally consist of 32 binary digits, so that two of them can be added in 32 microseconds. Likewise we shall do multiplications in the form of a number of consecutive additions of one and one or one and zero etc. There are 1024 such additions or thereabouts to be done in a multiplication of one 32 digit number by another, so that one might expect a multiplication to take about a millisecond. Actually the multiplier to be used on ACE will take rather over two milliseconds. This may sound rather long, when the unit operation is only a microsecond, but it actually seems that the machine is fairly well balanced in this respect, Le. the multiplication time is not a serious bottleneck. Computers always spend just as long in writing numbers down and deciding what to do next as they do in actual multiplications, and it is just the same with the ACE. A great deal of time is spent in getting numbers in and out of storage and deciding what to do next. To complete the four elementary processes, subtraction is done by complementation and addition, and division is done by the use of the iteration formula

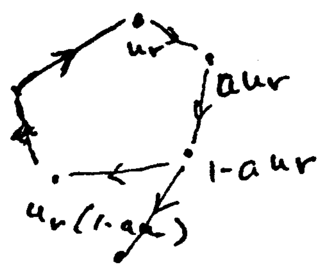
seminarTEXT

$$u_n = u_{n-1} + u_{n-1}(1 - au_{n-1})$$

u_n converges to a^{-1} provided $|1 - au_0| < 1$. The error is squared at each step, so that the convergence is very rapid. This process is of course programmed, i.e. the only extra apparatus required is the delay lines required for storing the relevant instructions.

Passing on from the arithmetic part there remains the input and output. For this purpose we have chosen Hollerith card equipment. We are able to obtain this without having to do any special development work. The speeds obtainable are not very impressive compared with the speeds at which the electronic equipment works, but they are quite sufficient in all cases where the calculation is long and the result concise: the interesting cases in fact. It might appear that there would be a difficulty in converting the information provided at the slow speeds appropriate to the Hollerith equipment to the high speeds required with the ACE, but it is really quite easy. The Hollerith speeds are so slow as to be counted zero or stop for many purposes, and the problem reduces to the simple one of converting a number of statically given digits into a stream of pulses. This can be done by means of a form of electronic commutator.

Before leaving the outline of the description of the machine I should mention some of the tactical situations that are met with in programming. I can illustrate two of them in connection with the calculation of the reciprocal described above. One of these is the idea of the iterative cycle. Each time that we go from u_r to u_{r+1}, we apply the same sequence of operations, and it will therefore be economical in storage space if we use the same instructions. Thus we go round and round a cycle of instructions:



It looks however as if we were in danger of getting stuck in this cycle, and unable to get out. The solution of this difficulty involves another tactical idea, that of 'discrimination' Le. of deciding what to do next partly according to the results of the machine itself, instead of according to data available to the programmer. In this case we include a discrimination in each cycle, which takes us out of the cycle when the value of $|1 - au|$ is sufficiently small. It is like an aeroplane circling over an aerodrome, and asking permission to land after each circle. This is a very simple idea, but is of the utmost importance. The idea of the iterative cycle of instructions will also be seen to be rather fundamental when it is realised that the majority of the instructions in the memory must be obeyed a great number of times. lf the whole memory were occupied by instructions, none of it being used for numbers or other data, and if each instruction were obeyed once only, but took the longest possible time, the machine could only remain working for sixteen seconds.

Another important idea is that of constructing an instruction and then obeying it. This can be used amongst other things for discrimination. In the example I have

seminarTEXT

just taken for instance we could calculate a quantity which was 1 if $|1 - au|$ was less than $2^{-3.1}$ and 0 otherwise. By adding this quantity to the instruction that is obeyed at the forking point the instruction can be completely altered in its effect when finally 1 - *au is* reduced to sufficiently small dimensions.

Probably the most important idea involved in instruction tables is that of standard *subsidiary tables*. Certain processes are used repeatedly in all sorts of different connections, and we wish to use the same instructions, from the same part of the memory every time. Thus we may use interpolation for the calculation of a great number of different fünctions, but we shall always use the same instruction table for interpolation. We have only to think out how this is to be done once, and forget then how it is done. Each time we want to do an interpolation we have only to remember the memory position where this table is kept, and make the appropriate reference in the instruction table which is using the interpolation. We might for instance be making up an instruction table for finding values of $J_0(x)$ and use the interpolation table in this way. We should then say that the interpolation table was a subsidiary to the table for calculating $J_0(x)$. There is thus a sort of hierarchy of tables. The interpolation table might be regarded as taking its orders from the J_0 table, and reporting its answers back to it. The master servant analogy is however not a very good one, as there are many more masters than servants, and many masters have to share the same servants.

Now let me give a picture of the operation of the machine. Let us begin with some problem which has been brought in by a customer. It will first go to the problems preparation section where it is examined to see whether it is in a suitable form and self-consistent, and a very rough computing procedure made out. lt then goes to the tables preparation section. Let us suppose for example that the problem was to tabulate solutions of the equation

$$y'' + xy' = J_0(x)$$

with initial conditions $x = y = 0$, $y' = a$. This would be regarded as a particular case of solving the equation

$$y'' = F(x, y, y')$$

for which one would have instruction tables already prepared. One would need also a table to produce the fünction $F(x, y, z)$ (in this case $F(x, y, z) = J_0(x) - xz$ which would mainly involve a table to produce $J_0(x)$, and this we might expect to get off the shelf). A few additional details about the boundary conditions and the length of the arc would have to be dealt with, but much of this detail would also be found on the shelf, just like the table for obtaining $J_0(x)$. The instructions for the job would therefore consist of a considerable number taken off the shelf together with a few made up specially for the job in question. The instruction cards for the standard processes would have already been punched, but the new ones would have to be done separately. When these had all been assembled and checked they would be taken to the input mechanism, which is simply a Hollerith card feed. They would be put into the card hopper and a button pressed to start the cards moving through. It must be remembered that initially there are no instructions in the machine, and one's normal facilities are therefore not available. The first few cards that pass in have therefore to be carefüllly thought out to deal with this situation. They are the initial input cards and are always the same. When they have passed in a few rather fundamental instruction tables will have been set up in the machine, including sufficient to enable the machine to read the special

seminarTEXT

pack of cards that has been prepared for the job we are doing. When this has been done there are various possibilities as to what happens next, depending on the way thejob has been programmed. The machine might have been made to go straight on through, and carry out the job, punching or printing all the answers required, and stopping when all of this has been done. But more probably it will have been arranged that the machine stops as soon as the instruction tables have been put in. This allows for the possibility of checking that the content of the memories is correct, and for a number of variations of procedure. lt is clearly a suitable moment for a break. We might also make a number of other breaks. For instance we might be interested in certain particular values of the parameter *a*, which were experimentally obtained figures, and it would then be convenient to pause after each parameter value, and feed the next parameter value in from another card. Or one might prefer to have the cards all ready in the hopper and let the ACE take them in as it wanted them. One can do as one wishes, but one must make up one's mind. Each time the machine pauses in this way a 'word' or sequence of 32 binary digits is displayed on neon bulbs. This word indicates the reason for stopping. I have already mentioned two possible reasons. A large class of further possible reasons is provided by the checks. The programming should be done in such a way that the ACE is frequently investigating identities which should be satisfied if all is as it should be. Whenever one of these checks fails the machine stops and displays a word which describes what check has failed.

It will be seen that the possibilities as to what one may do are immense. One of our difficulties will be the maintainence of an appropriate discipline, so that we do not lose track of what we are doing. We shall need a number of efficient librarian types to keep us in order.

Finally I should like to make a few conjectures as to the repercussions that electronic digital computing machinery will have on mathematics. I have already mentioned that the ACE will do the work of about 10,000 computers. It is to be expected therefore that large scale hand-computing will die out. Computers will still be employed on small calculations, such as the substitution of values in formulae, but whenever a single calculation may be expected to take a human computer days of work, it will presumably be done by an electronic computer instead. This will not necessitate everyone interested in such work having an electronic computer. It would be quite possible to arrange to control a distant computer by means of a telephone line. Special input and output machinery would be developed for use at these out stations, and would cost a few hundred pounds at most. The main bulk of the work done by these computers will however consist of problems which could not have been tackled by hand computing because of the scale of the undertaking. In order to supply the machine with these problems we shall need a great number of mathematicians of ability. These mathematicians will be needed in order to do the preliminary research on the problems, putting them into a form for computation. There will be considerable scope for analysts. When a human computer is working on a problem he can usually apply some common sense to give him an idea of how accurate his answers are. With a digital computer we can no longer rely on common sense, and the bounds of error must be based on some proved inequalities. We need analysts to find the appropriate inequalities for us. The inequalities need not always be explicit, i.e. one need not have them in such a form that we can tell, before the calculation starts, and using only pencil and paper, how big the error will be. The error calculation may be a serious part of the ACE's duties. To an extent it may be possible to replace the estimates of

seminarTEXT

error by statistical estimates obtained by repeating the job several times, and doing the rounding off differently each time, controlling it by some random element, some electronic roulette wheel. Such statistical estimates however leave much in doubt, are wasteful in machine time, and give no indication of what can be done if it turns out that the errors are intolerably large. The statistical method can only help the analyst, not replace him.

Analysis is just one of the purposes for which we shall need good mathematicians. Roughly speaking those who work in connection with the ACE will be divided into its masters and its servants. Its masters will plan out instruction tables for it, thinking up deeper and deeper ways of using it. Its servants will feed it with cards as it calls for them. They will put right any parts that go wrong. They will assemble data that it requires. In fact the servants will take the place of limbs. As time goes on the calculator itself will take over the fünctions both of masters and of servants. The servants will be replaced by mechanical and electrical limbs and sense organs. One might for instance provide curve followers to enable data to be taken direct from curves instead of having girls read off values and punch them on cards. The masters are liable to get replaced because as soon as any technique becomes at all stereotyped it becomes possible to devise a system of instruction tables which will enable the electronic computer to do it for itself. It may happen however that the masters will refuse to do this. They may be unwilling to let their jobs be stolen from them in this way. In that case they would surround the whole of their work with mystery and make excuses, couched in well chosen gibberish, whenever any dangerous suggestions were made. I think that a reaction of this kind is a very real danger. This topic naturally leads to the question as to how far it is possible in principle for a computing machine to simulate human activities. I will return to this later, when I have discussed the effects of these machines on mathematics a little fürther.

I expect that digital computing machines will eventually stimulate a considerable interest in symbolie logic and mathematical philosophy. The language in which one communicates with these machines, Le. the language of instruction tables, forms a sort of symbolic logic. The machine interprets whatever it is told in a quite definite manner without any sense of humour or sense of proportion. Unless in communicating with it one says exactly what one means, trouble is bound to result. Actually one could communicate with these machines in any language provided it was an exact language, i.e. in principle one should be able to communicate in any symbolic logic, provided that the machine were given instruction tables which would enable it to interpret that logical system. This would mean that there will be much more practical scope for logical systems than there has been in the past. Some attempts will probably be made to get the machine to do actual manipulations of mathematical formulae. To do so will require the development of a special logical system for the purpose. This system should resemble normal mathematical procedure closely, but at the same time should be as unambiguous as possible. As regards mathematical philosophy, since the machines will be doing more and more mathematics themselves, the centre of gravity of the human interest will be driven fürther and fürther into philosophical questions of what can in principle be done etc.

It has been said that computing machines can only carry out the processes that they are instructed to do. This is certainly true in the sense that if they do something other than what they were instructed then they have just made some

seminarTEXT

mistake. It is also true that the intention in constructing these machines in the first instance is to treat them as slaves, giving them only jobs which have been thought out in detail, jobs such that the user of the machine füllly understands what in principle is going on all the time. Up till the present machines have only been used in this way. But is it necessary that they should always be used in such a manner? Let us suppose we have set up a machine with certain initial instruction tables, so constructed that these tables might on occasion, if good reason arose, modify those tables. One can imagine that after the machine had been operating for some time, the instructions would have altered out of all recognition, but nevertheless still be such that one would have to admit that the machine was still doing very worthwhile calculations. Possibly it might still be getting results of the type desired when the machine was first set up, but in a much more efficient manner. In such a case one would have to admit that the progress of the machine had not been foreseen when its original instructions were put in. It would be like a pupil who had learnt much from his master, but had added much more by his own work. When this happens I feel that one is obliged to regard the machine as showing intelligence. As soon as one can provide a reasonably large memory capacity it should be possible to begin to experiment on these lines. The memory capacity of the human brain is probably of the order of ten thousand million binary digits. But most of this is probably used in remembering visual impressions, and other comparatively wasteful ways. One might reasonably hope to be able to make some real progress with a few million digits, especially if one confined one's investigations to some rather limited field such as the game of chess. It would probably be quite easy to find instruction tables which would enable the ACE to win against an average player. Indeed Shannon of Bell Telephone laboratories tells me that he has won games playing by rule of thumb: the skill of his opponents is not stated. But I would not consider such a victory very significant. What we want is a machine that can learn from experience. The possibility of letting the machine alter its own instructions provides the mechanism for this, but this of course does not get us very far.

It might be argued that there is a fundamental contradiction in the idea of a machine with intelligence. It is certainly true that 'acting like a machine', has become synonymous with lack of adaptability. But the reason for this is obvious. Machines in the past have had very little storage, and there has been no question of the machine having any discretion. The argument might however be put into a more aggressive form. It has for instance been shown that with certain logical systems there can be no machine which will distinguish provable formulae of the system from unprovable, i.e. that there is no test that the machine can apply which will divide propositions with certainty into these two classes. Thus if a machine is made for this purpose it must in some cases fail to give an answer. On the other hand if a mathematician is confronted with such a problem he would search around and find new methods of proof, so that he ought eventually to be able to reach a decision about any given formula. This would be the argument. Against it I would say that fair play must be given to the machine. Instead of it sometimes giving no answer we could arrange that it gives occasional wrong answers. But the human mathematician would likewise make blunders when trying out new techniques. It is easy for us to regard these blunders as not counting and give him another chance, but the machine would probably be allowed no mercy. In other words then, if a machine is expected to be infallible, it cannot also be intelligent. There are several mathematical theorems which say almost exactly that. But these theorems say nothing about how much intelligence may be displayed if a machine

seminarTEXT

makes no pretence at infallibility. To continue my plea for 'fair play for the machines' when testing their I.Q. A human mathematician has always undergone an extensive training. This training may be regarded as not unlike putting instruction tables into a machine. One must therefore not expect a machine to do a very great deal of building up of instruction tables on its own. No man adds very much to the body of knowledge, why should we expect more of a machine? Putting the same point differently, the machine must be allowed to have contact with human beings in order that it may adapt itself to their standards. The game of chess may perhaps be rather suitable for this purpose, as the moves of the machine's opponent will automatically provide this contact.

seminarTEXT

Lecture to L.M.S. Feb. 20 1947

The automatic computing engine now being designed at N.P.L. is a typical large scale electronic digital computing machine. In a single ~~xxxxx~~ lecture it will not be possible to give much technical detail of this machine, and most of what I shall say will apply equally to any other machine of this type now being planned ~~xxxxxxxxxx~~.

From the point of view of the mathematician the property of being digital should be of greater interest than that of being electronic. That it is electronic is certainly important because ~~xxxxx~~ these machines owe their high speed to this, and without the ~~xxxx~~ speed it is doubtful if financial support ~~xxxx~~ for their construction would be forthcoming. But this is virtually all that there is to be said on that subject. That the machine is digital however has more subtle significance. It means firstly that numbers are represented by sequences of digits which can be as long as one wishes. One can therefore work to any desired degree of accuracy. This accuracy is not obtained by more careful machining of parts, control of temperature variations, and such means, but by a slight increase in the amount of equipment in the machine. ~~xxxx~~ To double the number of significant figures used would involve increasing the equipment by a factor definitely less than two, and would also have some effect in increasing the time taken over each job. This is in sharp contrast with analogue machines, and continuous variable machines such as the differential analyser, where each additional decimal digit required necessitates a complete redesign of the machine, and an increase in the cost by perhaps as much as a factor of 10.

A second advantage of digital computing machines is that they are not restricted in their applications to any particular type of problem. The differential analyser is by far the most general type of analogue machine yet produced, but even it is comparatively limited in its scope. It can be made to deal with almost any kind of ordinary differential equation, but it is hardly able to deal with partial differential equations at all, and certainly cannot manage large numbers of linear simultaneous equations, or the zeros of polynomials. With ~~xx~~ digital machines however it is almost literally true that they are able to tackle any computing problem. A good working rule is that the ACE can be made to do any job that could be done by a human computer, and will do it in one ten-thousandth of the time. This time estimate is fairly reliable, except in cases where the job is ~~xxxxxxx~~ too trivial ~~xxxxxxxxxxxxxx xxxxxxxxxxxxxxxxx~~ to be worth while giving to the ACE.

Some years ago I was researching on what might now be described as an investigation of the theoretical possibilities and limitations of digital computing machines. I considered a type of machine which had a central ~~xxxxxx~~ mechanism, and an infinite memory which was contained on an infinite tape. This type of machine appeared to be sufficiently general. One of my conclusions was that the idea of a 'rule of thumb' process and a 'machine process' were synonymous. The expression 'machine process' ~~here~~ of course means one which could be carried out by ~~my~~ the type of machine I was considering. It was essential in these theoretical arguments that the memory should be infinite. It can easily be shown that otherwise the machine can only execute periodic operations. Machines such as the ACE may be regarded as practical versions of this same type of machine. There is at least a very close analogy.

digital computing machines will have a central mechanism or control and some very extensive form of memory. The memory does not need to be infinite, but it certainly needs to be very large. In general the arrangement of the memory on an infinite tape is unsatisfactory in a practical machine, because of the large amount of time which is liable to be spent in shifting up and down the tape to reach the point at which a particular piece of information required at the moment is stored. Thus a problem might easily need a storage of three million entries, and if each entry was equally likely to be the next required the average journey up the tape would be through a million entries, and this would be intolerable. One needs some form of memory with which any required entry can be reached at short notice. This difficulty presumably used to worry the Egyptians when their books were written on papyrus scrolls. It must have been slow work looking up references in them, and the present ~~of~~ arrangement of written matter in books which can be opened at any point is greatly to be preferred. We may say that storage on tape and papyrus scrolls is somewhat inaccessible. It takes a considerable time to find a given entry. Memory in book form is a good deal better, and is certainly highly suitable when it is to be read by the human eye. We could even imagine a computing machine that was made to work with a memory based on books. It would not be very easy but would be immensely preferable to the ~~infinite~~ single long tape. Let us for the sake of argument suppose that the difficulties involved in using books as memory were overcome, that is to say that mechanical devices for finding the right book and opening it at the right ~~place~~ page, etc. etc. had been developed, imitating the use of human hands and eyes. The information contained in the books would still be rather inaccessible because of the time occupied in the mechanical motions. One cannot turn a page over very quickly without tearing it, ~~[illegible]~~ and if one were ~~as for instance in using a book~~

—

to do much book transportation, and do it fast the energy involved would be very great. Thus if we moved one book every millisecond and each was moved ten metres and weighed 200 grams, and if the kinetic energy were wasted each time we should consume 10^{10} watts, about half the country's power consumption. If we are to have a really fast machine then, we must have our information, or at any rate a part of it, in a more accessible form than can be obtained with books. It seems that this can only be done at the expense of compactness and economy, e.g. by cutting the pages out of the books, and putting each one in to a separate reading mechanism. Some of the methods of storage which are being developed at the present time are not unlike this.

If one wishes to go to the extreme of accessibility in storage mechanisms one is liable to find that it is gained at the price of an intolerable loss of compactness and economy. For instance the most accessible known form of storage is that provided by the valve flip-flop or Jordan Eccles trigger circuit. This enables us to store one digit, capable of two values, and uses two thermionic valves. To store the content of an ordinary novel by such means would cost many millions of pounds. We clearly need some compromise method of storage which is more accessible than paper, film etc, but more economical in space and money than the straightforward use of valves. Another desirable feature is that it should be possible to record into the memory from within the computing machine, and this should be possible whether or not the storage already contains something, i.e. the storage should be erasible.

There are three main types of storage which have been developed recently and have these properties in greater or less degree. Magnetic wire is very compact, is erasible, can be recorded on from within the machine, and is moderately accessible. There is storage in the form of charge patterns on the screen of a cathode ray tube. This is probably the ultimate solution. It should eventually be nearly as accessible as the Jordan Eccles circuit.

to transport a book every millisecond, and

A third possibility is provided by acoustic delay lines. They give greater accessibility than the magnetic wire, though less than the C.R.T type. The accessibility is adequate for most purposes. Their chief advantage is that they are already a going concern. It is intended that the main memory of the ACE shall be provided by acoustic delay lines, consisting of mercury tanks.

The idea of using acoustic delay lines as memory units is due I believe to Eckert of Philadelphia University, who was the ~~chief~~ engineer ~~mainly~~ chiefly responsible for the Eniac. The idea is to store the information ~~required~~ in the form of compression waves travelling along a column of mercury. Liquids and solids will transmit sound of surprisingly high frequency, and it is ~~is~~ quite feasible to put as many as 1000 pulses into a single 5' tube.

P.T.O

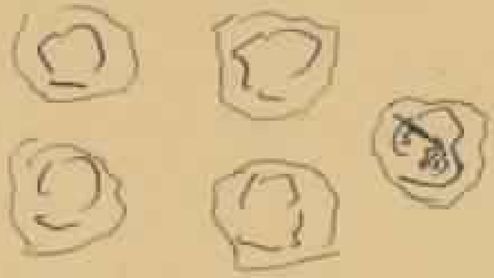
or the information what they represent
A train of pulses may be regarded as stored in the mercury whilst it is travelling through it. If the information is not required when the train emerges it ~~will~~ can be fed back into the column again and again until such time as it is required. This requires a 'recirculating circuit' to read the signal as it emerges from the tank and amplify it and feed it in again. If this were done with a simple amplifier it is clear that the characteristics of both the ~~tan~~ tank and the ~~ampl~~ amplifier would have to be extremely good to permit the signal to pass through even as many as ten times. Actually the recirculating circuit does something slightly different. What it does may perhaps be best expressed in terms of point set topology.

The signals may be conveyed into the mercury by a piezo-electric crystal, and also detected at the far end by another quartz crystal.

Let the plane of the diagram represent the space of all possible signals. I do not of course wish to imply that this is two dimensional. Let the function f be defined for arguments in this signal space and have values in it. In fact let f(s) represent the effect on the signal s when it is passed through the tank and the recirculating mechanism, *R.I.C.* Then a necessary and sufficient condition that the tank can be used as a storage which will distinguish between N different signals is that ~~we~~ there must be N sets $E_1 \ldots E_N$ such that, *if F_r is set of pts within distance ε of E_r*

$s \in F_r \supset f(s) \in E_r$
and the sets F_r are disjoint

~~xxxxxxx~~

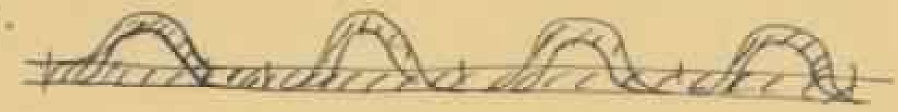


It is clearly sufficient for me here only then to ensure that the signals initially fed in belong to one or other of these sets *F_r* and it will remain in the set after any number of recirculations, without any danger of confusion. It is necessary for suppose $s_1 \ldots s_N$ are signals which can be fed into the machine at any time and read out later without fear of confusion. ~~Suppose that signals distant less than ε are liable to be confused. Then for each m, n, r, p $f^r(s_m)$ is distant more than ε from $f^p(s_n)$ provided $m \neq n$. We may then take E_m to be the set of points $f^r(s_m)$. These sets satisfy (x) and are distant ε at least apart.~~

Let E_r be the set of signals which could be obtained from s_r by successive applications of f and shifts of distance less than ε. Then the sets E_r are disjoint, since there are infinite, and by applying a shift of distance ε or less to pts of E_r we still

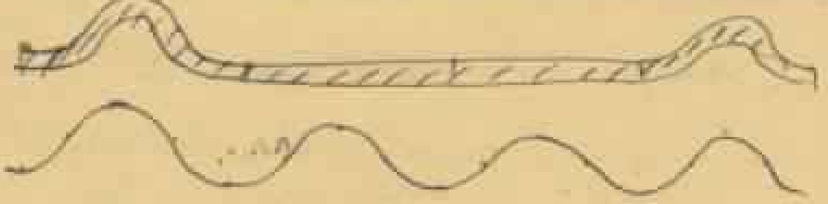
We assume however that owing to thermal agitation the effect of recirculation may be to give any pt within a circle of radius δ of $f(p)$

In the case of a mercury delay line used for $N \doteq 16$ the set would consist of all continuous signals within the shaded area.



One of the sets would consist of all continuous signals lying in the region below. It would represent the signal 1001.

~~In order to put such a recirculation system into effect it~~



In order to put such a recirculation system into effect it is essential that a clock signal be supplied to the memory system so that it will be able to distinguish the times when a pulse if any should be present. It would for instance be natural to supply a timing sine wave as shown above to the recirculator.

P.T.O.

The importance of a clock to the ~~recirculation~~ dynamic process in delay lines may be illustrated by an interesting little theorem. Suppose that instead of the condition $s \in \Sigma_r \supset f(s) \in \Sigma_r$ we impose a stronger one, viz $f^n(s) \to c_r$ if $s \in \Sigma_r$, i.e. there are ideal forms of the distinguishable signals, and each admissible signal converges towards the ideal form after recirculating. Then we can show that unless there is a clock the ideal signals are all constants. For let U_α represent a shift of origin, i.e. $U_\alpha s(t) = s(t+\alpha)$. Then since there is no clock the properties of the recirculator are the same at all times and f therefore commutes with U_α. Then $f U_\alpha(c_r) = U_\alpha f(c_r) = U_\alpha c_r$, for $f(c_r) = c_r$ since c_r is an ideal signal. But this means that $U_\alpha(c_r)$ is an ideal signal, and therefore for sufficiently small α must

The idea of a process f with the properties we have described is a very common one in connection with storage devices. It is known as 'regeneration' of storage. It is always present in some form, but sometimes the regeneration is as it were naturally occurring and ~~does not need to be~~ no precautions have to be taken. In other cases ~~it has to be~~ special precautions have to be taken to carry out such an f process or else the impression will fade.

be c_r, since the ideal signals are discrete. Then for any β and sufficiently large n, β/n will be sufficiently small and $U_{\beta/n}(c) = c$. But then by iteration $U^{n}_{\beta/n}(c) = U_{\beta}(c) = c$, i.e. $c(t+\beta) = c(t)$. This means that the ideal signal c is a constant.

We might say that the clock enables us to introduce a discreteness into time, so that time can for some purposes be regarded as a succession of instants instead of as a continuous flow. A digital machine must essentially deal with discrete objects, and in the case of the ACE this is made possible by the use of a clock. All other digital computing machines that I know of except for human and other brains do the same. One can think up ways of avoiding it, but they are very awkward. I should mention that the use of the clock in the ACE is not confined to the recirculation process, but is used in almost every part.

It may be as well to mention some figures connected with the mercury delay line as we shall use it. We shall use five foot tubes, with an inside diameter of half an inch. Each of these will enable us to store 1024 binary digits. The unit I have used here to describe storage capacity is self explanatory. A storage mechanism has a capacity of m binary digits if it can remember any sequence of m digits each being a 0 or a 1. The storage capacity is also the logarithm to the base 2 of the number of different signals which can be remembered, i.e. $\log_2 N$. The digits will be placed at a time interval of one microsecond, so that the time taken for the waves to travel down the tube is just over a millisecond. The velocity is about one and a half kilometres per second. The delay in accessibility time or average waiting for a given piece of information is about half a millisecond. In practice this is reduced to an effective 150 μs.

The full storage capacity of the ACE available on Hg delay lines will be about 200,000 binary digits. This is probably comparable with the memory capacity of a minnow.

I have spent a considerable time in this lecture on this question of memory, because I believe that the provision of proper storage is the key to the problem of the digital computer, and certainly if they are to be persuaded to show any sort of genuine intelligence much larger capacities than are yet available must be provided. In my opinion this problem of making a large memory available at reasonably short notice is much more important than that of doing operations such as multiplication at high speed. Speed is necessary if the machine is to work fast enough for the machine to be commercially valuable, but a large storage capacity is necessary if it is to be capable of anything more than rather trivial operations. The storage capacity is therefore the more fundamental requirement.

Let us now return to the analogy of the theoretical computing machines with an infinite tape. It can be shown that a single special machine of that type can be made to do the work of all. It could in fact be made to work as a model of any other machine.
is
~~[illegible]~~
The special machine may be called the universal machine; it works in the following quite simple manner. When we have decided what machine we wish to imitate we punch a description of it on the tape of the universal machine. This description explains what the machine would do in every configuration in which it might find itself. The universal machine has only to keep looking at this description in order to find out what it should do at each stage. Thus the complexity of the machine to be imitated is concentrated in the tape and does not appear in the universal machine proper in any way.

If we take the properties of the universal machine in ~~combination~~ combination with the fact that machine processes and rule of thumb processes are synonymous we may say that the universal machine is one which, when supplied with the appropriate instructions, can be made to do any rule of thumb process. This feature is paralleled in digital computing machines such as the ACE. They are in fact practical versions of the universal machine. There is a certain central pool of electronic equipment, and a large memory. When any particular problem has to be handled the appropriate instructions for the computing process involved are stored in the memory of the ACE and it is then 'set up' for carrying out that process.

I have ~~xxx~~ indicated the main strategic ideas behind digital computing machinery, and will now follow this account up with the very briefest description of the ACE. It may be divided for the sake of argument into the following parts

Memory

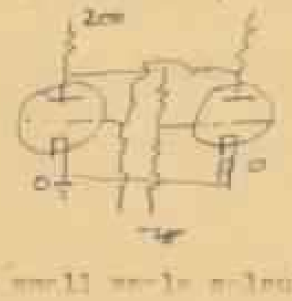
Control

Arithmetic part

Input and Output

I have already said enough about the memory and will only repeat that in the ACE the memory will consist mainly of 200 mercury delay lines each holding 1024 binary digits. The purpose of the control is to take the right instructions from the memory, see what ~~it~~ they mean, and arrange for them to be carried out. It is understood that a certain 'code of instructions' has been laid down, whereby ~~xxxxxx~~ each 'word' or combinations of say 32 binary digits ~~xxx~~ describes some particular operation. The circuit of the control is made in accordance with the code, so that the right effect is produced. To a large extent we have also allowed the circuit to determine the code, i.e. we have not just thought up ~~the~~ an imaginary 'best code' and then found a circuit to put it into effect, but have often ~~xxxxx~~ simplified the circuit at the expense of the code. It is also quite difficult to think about the code entirely in abstracto without any kind of circuit.

The arithmetic part of the machine is the part concerned with addition, multiplication and any other operations which it seems worth while to do by means of special circuits rather than through the simple facilities provided by the control. The distinction between control and arithmetic part is a rather hazy one, but at any rate it is clear that the machine should at least have an adder and a multiplier, even if they turn out in the end to be part of the control. This is the point at which I should mention that the machine is operated in the binary scale, with two qualifications. Inputs from externally provided data are in decimal, and so are outputs intended for human eyes rather than for later reconsumption by the ACE. This is the first qualification. The second is that, in spite of the intention of binary working there can be no bar on decimal working of a kind, because of the relation of the ACE to the universal machine. Binary working is the most natural thing to do with any large scale computer. It is much easier to work in the scale of two than any other, because it is so easy to produce mechanisms which have two positions of stability: the two positions may then be regarded as representing 0 and 1. Examples are lever as diagram, Jordan Eccles circuit, thyratron



If one is concerned with a small scale calculating machine then there is at least one serious objection to binary working.

For practical use it will be necessary to build a converter to transform numbers from the binary form to the decimal and back. This may well be a larger undertaking than the binary calculator. With the large scale machines this argument carries no weight. In the first place a converter would become a relatively small piece of apparatus, and in the second it would not really be necessary. This last statement sounds quite paradoxical, but it is a simple consequence of the fact that these machines can be made to do any rule of thumb process by remembering suitable instructions. In particular it can be made to do binary decimal conversion. For example in the case of the ACE the provision of the converter involves no more than adding two extra delay lines to the memory. This situation is very typical of what happens with the ACE. There are many fussy little details which have to be taken care of, and which, according to normal engineering practice would require special circuits. We are able to deal with these points without modification of the machine itself, by pure paper work, eventually resulting in feeding in appropriate instruction cards.

To return to the various parts of the machine. I was saying that it will work in the scale of two. It is not unnatural to use the convention that an electrical pulse shall represent the digit 1 and that absence of a pulse shall represent a digit 0. Thus a sequence of digits 0010110 would be represented by a signal like

where the time interval might be one microsecond.

Let us now look at what the process of binary addition is like. In ordinary decimal addition we always begin from the right, and the same naturally applies in binary. We have to do this because we cannot tell ~~xxxxxx~~ whether to carry unless we have already dealt with the less significant columns. The same applies with electronic addition, and therefore it is convenient to use the convention that if a sequence of pulses is coming down a line, then the least significant pulse always comes first. This has the unfortunate result that we must either write the least significant digit on the left in our binary numbers or else make time flow from right to left in our diagrams. As the latter alternative would involve writing from right to left as well as adding in that way, we have decided to put the least significant digit on the left. Now let us do a typical addition. Let us write the carry digits above the addends.

Carry	0 1 1 1 1 1 1 0 0 1 1
A	0 1 1 0 1 1 0 0 1 0 1 ...
B	0 1 1 1 0 1 0 0 1 1 ...
	0 1 0 0 1 1 1 0 0 0

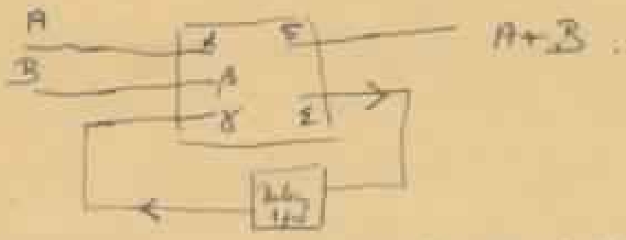
Note that I can do the addition only looking at a small part of the data. To do the addition electronically we need to produce a circuit with three inputs and two outputs.

Inputs		Outputs	
Addend A	α	Sum	ς
Addend B	β	Carry	ε.
Carry from last column	γ		

This circuit must be such that

if no. of 1's on inputs α, β, γ is	0	Then sum	0	and		0
	1	Σ	1	carry		0
	2	is	0	c		1
	3		1	is		1

~~[illegible]~~ It is very easy to produce a voltage proportional to the number of pulses on the inputs, and one then merely has to provide a circuit which will discriminate between four different levels and put out the appropriate sum and carry digits. I will not attempt to describe such a circuit; ~~[illegible]~~ it can be quite simple. When we are given the circuit we merely have to connect it up with feedback and it is an adder. Thus:-



It will be seen that we have made use of the fact that the same process is used in addition with each digit, and ~~[illegible]~~ and also the fact that the properties of the electrical circuit are invariant under time shifts, at any rate if these are multiples of the clock period. It might be said that we have made use of the isomorphism between the group of these time shifts and the ~~[illegible]~~ multiplicative group of real numbers to simplify our apparatus, though I doubt if many other applications of this principle could be found.

It will be seen that with such an adder the addition is broken down into the most elementary steps possible, such as adding one and one. Each of these occupies a microsecond. Our numbers will normally consist of 32 binary digits, so that two of them can be added in 32 microseconds. Likewise we shall do multiplications in the form of a number of consecutive additions of one and one or one and zero etc.

There are 1024 such additions or thereabouts to be done in a multiplication of one 32 digit number by another, so that one might expect a multiplication to take about a millisecond. Actually the multiplier to be used on ACE will take rather over two milliseconds. This may sound rather long, when the unit operation is only a microsecond, but it actually seems that the machine is fairly well balanced in this respect, i.e. the multiplication time is not a ~~xxxxxx~~ serious bottleneck. ~~xxxxxxx x~~ Computers always spend just as long in writing numbers down and deciding what to do next as they do in actual multiplications, and it is just the same with the ACE. A great deal of time is spent in getting numbers in and out of storage and deciding what to do next. To complete the four elementary processes, subtraction is done by complementation and addition, and division is done by the use of the iteration formula

$$u_n = u_{n-1} + u_{n-1}(1 - a u_{n-1})$$

u_n ~~which~~ converges to a^{-1} provided $|1 - a u_0| < 1$. The error is squared at each step, so that the convergence is very rapid. This process is of course programmed, i.e. the only extra apparatus required is the delay lines required for storing the relevant instructions.

Passing on from the arithmetic part there remains the input and output. For this purpose we have chosen Hollerith card equipment. We are able to obtain this without ~~xxx~~ having to do any special development work. The speeds obtainable are not very impressive compared with the speeds at which the electronic equipment works, but they are quite sufficient ~~xxxxx~~ in all cases where the calculation is long and the result concise: the interesting cases in fact.

It might be supposed that there would be a difficulty in converting the information provided at the slow speeds appropriate to the Hollerith equipment to the high speeds required with the ACE, but it is really quite easy. The Hollerith speeds are so slow as to be counted zero or stop, for many purposes, and the problem reduces to the simple one of converting a ~~xxx~~ number of statically given digits into a stream of pulses. This can be done by means of a form of electronic commutator.

16a and → 16 b.

Now let me give a picture of the operation of the machine. Let us begin with some problem which has been brought in by a customer. It will first go to the problems preparation section where it is examined to see whether it is in a suitable form and self-consistent, and a very rough computing procedure made out. It then goes to the tables preparation section. ~~xxx~~ Let us suppose for example that the problem was to ~~find~~ tabulate solutions of the equation

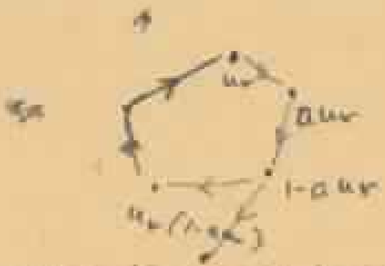
$$y'' + xy' = J_0(x)$$

with initial conditions $x=y=0$, $y'=a$. This would be regarded as a particular case of solving the equation

$$y'' = F(x, y, y')$$

for which one would have instruction tables already prepared. One would need also a table to produce the function $F(x,y,z)$ (in this case $F(x,y,z) = J_0(x) - xz$) which would mainly involve a table to produce $J_0(x)$, and this we might expect to get off the shelf). A few additional details about the boundary conditions and the ~~numerical digits~~ length of the arc would have to be dealt with, but much of this detail would also be found on the shelf, just like the table for obtaining $J_0(x)$. The instructions for the job would therefore consist of a considerable number taken off the shelf together with a few made up specially for the job in question. The instruction cards for the standard processes would have already been ~~made~~ punched, but the new ones would have to be done separately. Then

Before leaving the outline of the description of the machine I should mention ~~xxx particular~~ some of the tactical situations that are met with in programming. ~~[illegible]~~ I can illustrate two of them in connection with the ~~di~~ calculation of the reciprocal described above. One of these is the idea of the iterative cycle. Each time that we go from u_r to u_{r+1} we apply the same sequence of operations, and it will therefore be economical in storage space if we use the same instructions. Thus we go round and round a cycle of instructions:—



It looks however as if we were in danger of getting stuck in this cycle, and unable to get out. ~~This~~ The solution of this difficulty involves another tactical idea that of 'discrimination' i.e. of ~~[illegible]~~ deciding what to do next partly according to the results ~~of~~ that the machine itself, instead of according to data available to the ~~[illegible]~~ programmer. In this case we include a discrimination in each cycle, which takes us out of the cycle when the value of $|1-au|$ is sufficiently small. It is like an aeroplane circling over an aerodrome, and asking permission to land after each circle. This is a very simple idea, but is of the utmost importance. The idea of the iterative cycle of instructions will also been seen to be rather fundamental when it is realised that the majority of the instructions in the ~~machine~~ memory must be obeyed a great number of times. If ~~xxx~~ the whole memory were occupied by instructions, none of it being used for numbers or other data, and if each instruction were obeyed once only, but took the longest possible time, the machine could only remain working for sixteen seconds.

Another important idea is that of constructing an instruction and then obeying it. This can be used amongst other things for discrimination. In the example ~~shown~~ I have just taken for instance we could ~~arrange~~ calculate a quantity which was 1 if $|1-au|$ was less than 2^{-31} and 0 otherwise. By adding this quantity to the instruction that is obeyed at the forking point the instruction can be completely altered in its effect when finally $1-au$ is reduced to sufficiently small dimensions.

Probably the most important idea involved in instruction tables is that of standard subsidiary tables. Certain processes are used repeatedly in all sorts of different connections, and we wish to use the same instructions, from the same part of the memory every time. Thus we may use interpolation for the calculation of a great number of different functions, but we shall always use the same ~~interpolation formula~~ instruction table for interpolation. We have only to think out how this is to be done once, and forget then how it is done. Each time we want to do an interpolation we have only to remember the memory position where this table is kept, and make the appropriate reference in the instruction table which is using the interpolation. We might for instance be making up an instruction table for finding values of $J_0(x)$ and use the interpolation table in this way. We should then say that the interpolation table was a subsidiary to the table for calculating $J_0(x)$. There is thus a sort of hierarchy of tables. The interpolation table might be regarded as taking its orders from the J_0 table, and reporting its answers back to it. The master servant analogy is however not a very good one, as there are many more masters than servants, and many masters have to share the same servant.

Return to p. 16.

these had all been assembled and checked they would be taken to the input mechanism, which is simply a Hollerith card feed. They would be put intothe card hopper and a button pressed to start the cards moving through. It must be remembered that initially there are no instructions in the machine, and ones normal facilities are therefore not available. The first few cards that pass in have therefore to be carefully thought out to deal with this situation. They are the initial input cards and are always the same. When they have passed in a few fundamental instruction tables will have been set up in the machine, including sufficient to enable the machine to read the special pack of cards that has been prepared for the job we are doing. When this has been done there are various possibilities as to what happens next, depending on the way the job has been programmed. The machine might have been made to go straight on through, and carry out the job, punching or printing all the answers required, and stopping when all of this has been done. But more probably it will have been arranged that the machine stops as soon as the instruction tables have been put in. This allows for the possibility of checking that the content of the memories is correct, and for a number of variations of procedure. It is clearly a suitable moment for a break. We might also make a number of other breaks. For instance ~~txxx~~ we might be interested in certain particular values of the parameter a , ~~xxxxxx~~ which were experimentally obtained figures, and it would then be convenient to pause after each parameter value, and feed the next parameter value in from another card. Or one might prefer to have the cards all ready in the hopper and let the ACE take them in as it wanted them. One can do as one wishes, but one must make up one's mind. Each time the machine pauses in this way a 'word' or sequence of 32 binary digits is displayed on neon bulbs. This word indicates the reason for stopping. I have already mentioned two possible reasons. A large class of further possible reasons is provided by the checks. The programming should be done in such a way that the ACE is frequently investigating identities which should be satisfied if all is as it should be. Whenever one of these checks fails the machine stops and displays a word which describes what check has failed.

It will be seen that the possibilities as to what one may do are immense. One of our difficulties will be the maintenance of an appropriate discipline, so that we do not lose track of what we are doing. We shall need a number of efficient librarian types to keep us in order.

Finally I should like to make a few conjectures as to the repercussions that electronic digital computing machinery will have on mathematics. I have already mentioned that the ACE will do the work of about 10,000 computers. It is to be expected therefore that large scale hand-computing will die out. Computers will still be ~~xxxx~~ employed on small calculations, such as the substitution of values in formulae, ~~xxxxxxxx~~ but whenever a single calculation may be expected to take a human computer days of work, it will presumably be done by an electronic computer instead. This will not necessitate everyone ~~x~~ interested in such work having an electronic computer. It would be quite possible to arrange to control a distant computer by means of a telephone line. ~~xxxxx~~ Special input and output machinery would be developed for use at these out stations, and would cost ~~xxxxxx~~ a few hundred pounds at most. ~~xxxxxxxxx xxxxxxxxxxxxxxxxxxx~~ The main bulk ~~xxxxxxx~~ of the work done by these computers will however consist of problems which could not have been tackled by hand computing because of the scale of the undertaking. In order to supply the machine with these problems we shall need a great ~~xxxxxxxxxxxxxxxxxxxxx~~ number of mathematicians of ~~xxx~~ ability. These mathematicians will be needed in order to do the preliminary research on the problems, putting them into a form for computation. There will be considerable scope for analysts. When a human computer is working on a problem he can usually apply some common sense to give him an idea of how accurate his answers are. With a digital computer ~~xxxxxxxxxx xxxx~~ we can no longer rely on common sense, and the bounds of error must be based on some proved inequalities. We need analysts to find the appropriate inequalities for us. The inequalities need not always be explicit, i.e. one need not have them in such a form that we can tell, before the calculation starts and using only pencil and paper, how big the error will be. The error calculations may be a serious part of the ACE's duties. To an extent it may be possible to replace the estimates of error by statistical estimates obtained by repeating the job several times, and doing the rounding off differently each time, controlling it by some random element, some electronic roulette wheel. Such statistical estimates however leave much in doubt, are wasteful in machine time, and give no indication of what can be done if it turns out that the errors are intolerably large. The statistical method can only help the analyst, not replace him.

Analysis is just one of the purposes for which we shall need good mathematicians. Roughly speaking those who work in connection with the ACE will be divided into its masters and its servants. Its masters will plan out instruction tables for it, thinking up deeper and deeper ways of using it. Its servants will feed it with cards as it calls for them. They will put right any parts that go wrong. They will assemble data that it requires. In fact the servants will take the place of limbs. As time goes on the calculator itself will take over the functions both of masters and of servants. The servants will be replaced by mechanical and electrical limbs and sense organs. One might for instance provide curve followers to enable data to be taken direct from curves instead of having girls read off values and punch them on cards. The masters are liable to get replaced because as soon as any technique becomes at all stereotyped it becomes possible to devise a system of instruction tables which will enable the electronic computer to do it for itself. It may happen however that the masters will refuse to do this. They may be unwilling to let their jobs be stolen from them in this way. In that case they would surround the whole of their work with mystery and make excuses, couched in well chosen gibberish, whenever any dangerous suggestions were made. I think that a reaction of this kind is a very real danger. This topic naturally leads to the question as to how far it is possible in principle for a computing machine to simulate human activities. I will return to this later, when I have discussed the effects of these machines on mathematics a little further.

I expect that digital computing machines will eventually stimulate a considerable interest in symbolic logic and mathematical philosophy. The language in which one communicates with these machines, i.e. the language of instruction tables, forms a sort of symbolic logic. The machine interprets whatever it is told in a quite definite manner without any sense of humour or sense of proportion. Unless in communicating with it one says exactly what one means, trouble is bound to result. Actually one could communicate with these machines in any language provided it was an exact language, i.e. in principle one should be able to communicate in any symbolic logic, provided that the machine were given instruction tables which would enable it to interpret that logical system. This should mean that there will be much more practical scope for logical systems than there has been in the past. p. 10 As regards mathematical philosophy, since the machines will be doing more and more mathematics themselves, the centre of gravity of the human interest will be driven further and further into philosophical questions of what can in principle be done etc.

Some attempts will probably be made to get the machines to do actual manipulations of mathematical formulae. ~~This~~ To do so will require the development of a special logical system for the purpose. This system should resemble normal mathematical procedure as closely ~~as possible~~, but at the same time should be as unambiguous as possible.

It has been said that computing machines can only carry out the processes that they are instructed to do. This is certainly true in the sense that if they do something other than what they were instructed then they have just made some mistake. It is also true that the intention in constructing these machines in the first instance is to treat them as slaves, giving them only jobs which have been thought out in detail, jobs such that the user of the machine fully understands in principle what is going on all the time. Up till the present machines have only been used in this way. But is it necessary that they should always be used in such a manner? Let us suppose we have set up a machine with certain initial instruction tables, so constructed that these tables might on occasion, if good reason arose, modify those tables. One can imagine that after the machine had been operating for some time, the instructions would have altered out of all recognition, but nevertheless still be such that one would have to admit that the machine was still doing very worthwhile calculations. Possibly it might still be getting results of the type desired when the machine was first set up, but in a much more efficient manner. In such a case one would have to admit that the progress of the machine had not been foreseen when its original instructions were put in. It would be like a pupil who had learnt much from his master, but had added much more by his own work. When this happens I feel that one is obliged to regard the machine as showing intelligence. As soon as one can provide a reasonably large memory capacity it should be possible to begin to experiment on these lines. The memory capacity of the human brain is probably of the order of ten thousand million binary digits. But most of this is probably used in remembering visual impressions, and other comparatively wasteful ways. One might reasonably hope to be able to make some real progress with a few million digits, especially if one confined one's investigations to some rather limited field such as the game of chess. It would probably be quite easy to find instruction tables which would enable the ACE to win against an average player. Indeed Shannon of Bell Telephone laboratories tells me that he has won games playing by rule of thumb: the skill of his opponents is not stated. But I would not consider such a victory very significant. What we want is a machine that can learn from experience. The possibility of letting the machine alter its own instructions provides the mechanism for this, but this of course does not get us very far.

It might be argued that there is a fundamental contradiction in the idea of a machine with intelligence. It is certainly true that 'acting like a machine', has become synonymous with lack of adaptability. But the reason for this is obvious. Machines in the past have had very little storage, and there has been no question of the machine having any discretion. The argument might however be put into a more aggressive form. It has for instance been shown that with certain logical systems there can be no machine which will distinguish provable formulae of the system from unprovable, i.e. that there is no test that the machine can apply which will divide propositions with certainty into these two classes. Thus if a machine is made for this purpose it must in some cases fail to give an answer. On the other hand if a mathematician is confronted with such a problem he would search around a find new methods of proof, so that he ought eventually to be able to reach a decision about any given formula. This would be the argument. Against it I would say that fair play must be given to the machine. Instead of it sometimes giving no answer we could arrange that it gives occasional wrong answers. But the human mathematician would likewise make blunders when trying out new techniques. It is easy for us to regard these blunders as not counting and give him another chance, but the machine would probably be allowed no mercy. In other words then, if a machine is expected to be infallible, it cannot also be intelligent. There are several mathematical theorems which say almost exactly that. But these theorems say nothing about how much intelligence may be displayed if a machine makes no pretence at ~~infallibil~~ infallibility. To continue ~~the~~ my plea for 'fair play for the machines' when testing their I.Q. A human mathematician ~~xxxxxxxxx~~ has always undergone an extensive training. This training may be regarded as not unlike putting instruction tables into a machine. One must therefore not expect a machine to do a very great deal of building up of instruction tables on its own. No man adds very much to the body of knowledge, why should we expect more of a machine? Putting the same point differently, the machine must be allowed to have contact with human beings in order that it may adapt itself to their standards. The game of chess may perhaps be rather suitable for this purpose, as the moves of the machine's opponent will automatically provide this contact.