on the use of DIGITAL COMPUTERS

*By **Philip M. Morse***

An invited address before the American Physical Society at the 25th Anniversary Meeting of the American Institute of Physics in New York, Feb. 1, 1956

Philip M. Morse is professor of physics at the Massachusetts Institute of Technology.

LET me say, right off, that I don't intend to give you a sales talk on how electronic computers will do all the observing for the experimental physicist and all the thinking for the theoretical physicist. That doesn't mean I think computers won't be of any use to us, however. They are tools, like cyclotrons and spectroscopes, which can help us carry on research but, as with any other tool, it's going to take a great deal of thought and ingenuity to realize their full potentialities. As with any of our other instruments, only those of us who take the time and trouble to learn thoroughly the computer's operations and its limitations will be able to exploit fully its potentialities.

Nor am I going to spend my time describing the gadgetry which is used to implement its various functions. Many of you could do that job better than I could and, besides, it is the operating pattern of the computer which is important for the user to understand, not so much the details of its realization in any particular computer.

The modern digital computer performs a number of familiar functions in an unfamiliar manner. It has an input, where the procedural instructions and values of the initial constants for a specific problem are translated into the specialized language understood by the machine. It has a storage unit or units, where the translated input, the results of intermediate steps, and, finally, the answer or answers are recorded, ready for later use. It has an arithmetic unit, which processes the numbers according to instructions. It has an output, so that the results of the prescheduled operations may be retranslated into more usual form and displayed in an appropriate and understandable manner. And, of course, it has a control unit, which coordinates and synchronizes all these activities.

None of these functions are essentially new to us. A desk computer is an arithmetic unit. The intermediate storage in most computing problems is a set of numbers on a piece of paper and the usual output is a table of numbers, a graph, or some other pictorial representation. Those of us who have had an assistant carry out a computation have also written out instructions similar in purpose to the coded programs for a machine. The major difference, aside from the much more impressive appearance of the new implementation, is the increase in speed and reliability. Present day digital computers can perform individual operations, such as adding or multiplying two eight-decimal-digit numbers or withdrawing such a number from or inserting it into a storage register, in a time of the order of magnitude of ten microseconds. And the magnetic core storage in Whirlwind I, the computer we have at Massachusetts Institute of Technology, for example, cannot now be blamed for more than one error in about 10^{10} demands on its memory!

Although there is no essential change in function, these very large improvements in speed and reliability make possible the solution of problems and the use of numerical methods which would be impractical by earlier means. A computation technique which involves a hundred or a thousand or even ten thousand trial calculations to find the answer can now be employed in a routine manner.

An important limitation on the reliability of complicated calculations is the number of times the storage unit is used. Calculations usually proceed in a sequence of steps; at the beginning of each step numbers are read from storage, at its end the results are put in storage. With hand computation, where the storage is numbers copied by hand onto a sheet of paper, the errors of transcription or of reading are about one in a hundred for the amateur, maybe as low as one in 10^4 for a professional. Consequently, if a computation involves a hundred or more steps, the reliability of a hand computation by a nonprofessional is not very high and, if the time required is so great as to preclude a number of complete repetitions, it's usually not worth carrying out the calculation. It could be carried out by machine, however, for the reliability of the answer would be very much greater and it would involve very much less time to repeat the calculations for a check. In addition, it is possible to program internal checks, which can spot many possible errors as the work progresses.

This reliability, especially in the storage unit, has only recently been achieved; at present the magnetic core storage units are superior, but ferroelectric devices might eventually compete or surpass them. Ordinary vacuum tubes make up the usual arithmetic unit; transistors will certainly help to reduce size and power requirements, when they can be used without sacrifice of speed or reliability.

But I said I didn't intend to boast of future possibilities. What I want to do is to discuss some of the shortcomings of present digital computers and then, by means of a series of examples, each with a moral attached to it, to indicate some of the ways we can exploit the computer's good points and obviate some of its deficiencies. I'll spend my time, as I have so far, discussing the behavior of the really big machines, the megabuck jobs, for here both advantages and shortcomings are more apparent.

THE major shortcomings of today's electronic calculators are in part instrumental and in part procedural. In the first place, the arithmetic and the storage portions of the computer have outstripped the other parts, the input and output units, in reliability, speed, and versatility. The input unit—the link between the written instructions from the person desiring the solution and the sequence of signals in the internal storage which tells the machine how to produce the solution—is less satisfactory. It is relatively slow and complicated and usually requires transposing the original instructions to card or tape by human means. Each such transposition may produce errors, errors not easy to guard against by internal checks.

This lack of simplicity and flexibility in the input equipment is a more serious handicap when it comes to using the computer to process experimental data. Many large-scale installations for experimental research already have special-purpose computers attached. But if the connection between the electrical or optical measuring device and the punched tape or card could be simplified, a great deal of the reduction and statistical analysis of data could be carried out on a general-purpose computer in a routine manner. It is to be hoped that more versatile input equipment will be developed in the near future, which will allow these potentialities to be realized.

Still more inadequate, as compared to the central units of the machine, is the output equipment. If the results can be given in terms of relatively few numbers, the present output is not too unsatisfactory; the result can be displayed on a cathode-ray scope or punched or printed on a card almost as quickly as the machine can carry out a thousand or so elementary operations. But if the results involve large lists of numbers, we reach the absurd situation where it takes much more time to get the answer out of the machine than it does for the machine to compute the answer in the first place.

We can often free the rest of the machine for other work, while this slow transcription is taking place, by employing a sequential procedure. We first run the answer out on magnetic tape and then, on an auxiliary machine, go from magnetic tape to punched paper tape or card and this, then, finally goes to the tabulator or automatic typewriter, which prints a table we can use. The disadvantages of this multiple sequence of operations in inconvenience and loss of reliability are obvious. In some machines one can go from magnetic tape direct to printer, but any machine involving mechanical printing is slow compared to the rest of the computer.

It is also possible to program the machine so it will print the answer in digital form on the surface of a cathode-ray tube, where it can be photographed at a rate of about one page of figures per second. This at least has the advantage of directness of path from the inside of the machine to the pages of numbers, but the photographic copies produced by this method at present fall far short of any reasonable standards of legibility and typography which one might place on numerical tables for general use. Future developments probably will improve this procedure, both as to speed and clarity. But at present, if one wishes to use the results more than a few times, the photographs are usually transcribed by hand to a more legible form, which procedure, of course, introduces possible errors and takes additional time.

The lack of flexibility, speed, and clarity of the output unit is still more apparent when we wish to display the answer in nondigital form, such as a graph or diagram. A well-drawn graph should be readable to one thousandth of the full range of scale, giving three significant decimal digits. A graph on the face of the usual cathode-ray tube, however, can hardly be read to better than two significant digits. Certainly a great deal of work is required before the output equipment begins to approach the present storage and arithmetic units in speed, reliability, and versatility.

These deficiencies are not too serious, however. Many of them will be remedied in the near future and most of them are not noticed by the occasional user of the machine, are not very apparent unless one is asking the utmost from the machine. A much more serious disability is not so easy to overcome, for it is inherent in the very nature of the machine; to ask that it be done away with is like asking to have the fun of swimming without the nuisance of getting wet. This bottleneck to rapid machine use, this barrier which every new user must learn to surmount, is the process of programming itself.

YOU see, although the central parts of the really fast machine, the high-speed storage and arithmetic units, are remarkably rapid and reliable, they are incapable of doing anything they are not told. And they must be told in complete and exhausting detail. When programming a calculation to be done by hand, one can assume that the computer, even though not a professional, will exercise a certain amount of judgment at various points along the way, will make the proper modifications if some number turns out to be negative and will call for help if things seem to be going wrong. To keep the programming simple, one often programs the work in sections, carrying a long computation up to an intermediate point, to see what the intermediate results are like, before deciding on the program details for the next stage.

Such casual programming techniques will never do for the really fast computers. The machine has to be told exactly what to do at each step in the calculation; what to do if the number turns out to be negative, what to do if the number is larger than the storage register it is supposed to fit into, not to go on computing terms in a series if they are negligibly small, not to try to divide by zero and so on and so on, in complete and exhausting detail. The important thing to remember in machine programming is that machine computers are dumber than anybody.

The comforting technique of doing the work in sections, and pausing to decide what to do next turns out to be quite inefficient because of the extreme disparity in the times for computing and those used for pausing. With hand computation it might take six hours to complete one section of the calculations and then take a half-hour to program the next step; with the machine it might take six seconds to complete the calculations and, because of the increased detail required, it might take two hours to program the next step. One obviously can't keep the machine waiting for those two hours, so the machine is switched to other problems and one has to wait in line to get back on, after the next section is programmed. If possible, therefore, it is best to program a problem in one continuous unit, bringing out of the machine only those final results which are necessary to know in the end. Because of the present relative inefficiency in output equipment, the fewer the numbers brought out of the machine, the quicker the results will be available.

Thus is posed a dilemma which can annoy and frustrate the novice. The final, working program should minimize the numbers brought out of the machine. But if the program has an error in it, or has not allowed for all possible alternatives at each step, the numbers coming out of the machine will be inconsistent nonsense and one must start "trouble-shooting". Or, even worse, the results may only be slightly inconsistent, and one is not sure whether there is an error or not. And here the lack of output details is a handicap, for one often has to examine the intermediate results for a number of trial computations before the programming error is found—and these intermediate results have not been brought out of the machine. There are, of course, so-called "post-mortem" techniques for bringing the machine contents to light after a bad run, so the faulty parts of the program can be diagnosed. And one soon learns the tricks to be used to write a "clean" program, one which is easy to diagnose and check. But it should be evident that trouble shooting a complex program can be as time consuming as trouble-shooting experimental equipment often is.

At present, the writing of a program for a large computational problem, checking it out and determining the accuracy of the final answers, is the biggest bottleneck of all in the use of the really fast computers. At MIT, for example, we have adopted the rule of thumb that any computation which can be completed by hand with an expenditure of less than about three man-months of time, and which won't be repeated sooner than a year, should *not* be programmed for Whirlwind. We have found by experience that the answers to such problems can usually be obtained quicker by hand, because a new program, for a tough problem, can easily take more than three man-months to perfect.

This does not mean that one should turn big problems over to professional programmers to code, either. Many of the critical details of the program can only be effectively designed by someone who knows both the problem and the techniques of programming. It is advisable that the novice, who wishes to have a big problem solved by machine, should spend a couple of weeks learning basic programming techniques, and then should work out his first program together with an experienced professional, who can help him avoid the usual pitfalls and can suggest the use of already-tested subprograms for part of the work. After this first experience the user can carry on with only occasional help and advice, and in a short time he usually prefers to do his own programming. Several such amateurs, after a few practice runs, have taught our professionals a trick or two.

As a matter of fact, after machine operation gets to be more standardized and after we have gained more experience in machine use, this programming bottleneck should ease considerably. After all, from a general point of view, a program is a kind of mathematical notation, expressing, in symbolic form, the logical relationships between various mathematical steps in the solution of the problem. As with the usual mathematical symbols, a poor notation can greatly hinder us, a good notation can make it easy to work out the solution.

At present we are not very far along in developing this new brand of notation for programming. The basic symbols, such as addition, transfer to storage, and the like, are built into the machine. The more complex ones are collected in libraries of subroutines. When, for example, someone works out a program for computing the exponential function for any specified argument, or a way to solve a five-by-five secular determinant, this program can be written up, with comments and estimates of error, and placed in a library of subroutines. Then, if anyone else requires the value of an exponential, as a part of his calculation, this subroutine can be inserted in his larger program. Eventually some of the simpler and more popular subroutines can then be built into the machine so that, for example, the code *exp*

(2.501) in the program would set the machine busily computing the exponential of 2.501, so it can be used in the next stage of the computation.

BUT I hadn't intended to spend so much time talking generalities. I think a few specific cases will exemplify some of the points I've been trying to make and will illustrate a few of the ways whereby a high-speed digital computer can be used by physicists. I'll pick the examples in physics—though I would have had a wider choice of examples in engineering or meteorology or geophysics—out of the several thousand research computations carried out by Whirlwind in the past few years. Also, my examples are chosen from work done on Whirlwind I at MIT because I know this work more intimately. Quite similar examples could be drawn from the work of any of the other large machines now operating.

An example of a very simple problem, requiring little machine experience, was the solution of about 500 five-by-five secular equations for J. C. Slater and G. F. Koster of our solid-state theory group. In this case the program for solution had already been devised for a previous problem and needed only minor modifications to be used. The values of the matrix components were all computed out by hand and fed in successively to the machine, the 2500 roots successively appearing in the output. The whole computation of the roots took the machine about four hours to accomplish. Although this was a long time for the machine, because of the large input and output, it represented a substantial saving of time for the computers, since no appreciable time had been spent on programming. The accuracy and reliability of the answers was better than would have been obtained by hand, without a great deal of time spent on checking. Here the machine was used like a glorified desk computer, and little of its potentialities for combined programs was realized. The volume of the output was fairly large, but the results were utilized in further hand computations, so no great care had to be taken to obtain good typography or format for the output.

A more interesting and much more ambitious problem, done by E. H. Jacobsen, involved the calculation of the density of vibrational energy levels for the crystal lattice of copper, needed in order to compute the temperature diffuse scattering of x-rays from the elastic constants of the lattice. In this problem 30 000 three-by-three secular equations were solved. If all the matrix components had been hand computed and then fed in, and if all the 90 000 energy values had been fed out of the machine, the inefficiencies of the input-output units would have caused great delays. More important, such a mass of numbers could not have been held at one time in the internal storage, so the machine would have had to stop frequently to unload answers and load up new data, which would have stretched the job out to a completely unacceptable length of time. In addition, the hand analysis of 90 000 answers would have been a tedious and discouraging job, to say the least.

To shorten the time, several things were done, each of them adding complexity to the program but making it easier for the machine. In the first place the machine was asked to compute all 270 000 matrix components. The machine could compute one of these components much faster than a precalculated value could be fed into its memory from outside. In addition, the machine didn't have to remember a lot of them, it could compute each value as it was needed; only the general directions for computing the values had to be remembered. In the second place, the computed energy values were never withdrawn from the machine. What was needed was, not the values of each of the 90 000 levels, but information on how many such levels there were in a sequence of adjoining energy bands: in other words, the density of levels in various ranges of energy was the answer sought.

To record the level density, several hundred storage registers in the machine were set aside to serve as a sequence of energy bands, covering the energy range of interest. Each energy level, as it was computed, was then read as a register address, not as a number, and thereby routed to the storage register for that energy value. For example, if a computed level turned out to be 253.7, storage register 253, which was counting all levels between 253 and 254, was increased by one unit, whereas if the level were 177.2, one unit would be added to register 177. At the end of the computation the tally count in each of these registers was just the number of levels in the corresponding energy band; the 90 000 individual energy values had long since been forgotten and all that was taken out of the machine were these several hundred level density counts, which took very little time to record. I am sure you realize that the actual process was not as simple as I have described it here, but the basic principle was similar. A problem, at first sight too big for Whirlwind, was thus squeezed on by clever programming.

I don't know how long it would have taken to have computed by hand 270 000 matrix components, from them to have obtained 90 000 energy values by hand, and then to have counted up all the levels occurring in the various energy bands in a sequence covering the appropriate energy range. It probably would have taken several man-years. The times actually required forcefully illustrate the magnitude of the speed-up, and also the size of the programming bottleneck. It took Whirlwind only one hour and a half to do all the calculations, literally millions of them. *But* it took six man-months to work out the program, test it, and get it ready for the hour and a half run. However, Jacobsen, who worked the program out, started from scratch, without any previous experience with big computers. If he had started with as much experience as he now has, it probably would have taken him only three months to work out the program, but I doubt whether he could have done it in much less time. And I doubt whether a professional programmer, if he had been handed a set of written instructions, could have done it at all, no matter how voluminous the written instructions were. Such problems simply can't be done on a mail-order basis, so to speak.

Another example will show how programs are combined and telescoped together as we learn how to use the equipment. A number of years ago some of us computed, by hand, a set of energy values for atomic wave functions of simple analytic form, as functions of the parameters of the form. These tables enabled one, by use of the variational principle, to obtain reasonably accurate, analytic wave functions for the lower states of atoms in the first row of the periodic table. Partly for training purposes, partly because the tables still seemed to be useful, Arnold Tubis, of our group, recently recomputed these tables, using Whirlwind. He programmed the machine to compute kinetic energies, nuclear potential energies, interaction and exchange energies for *s*, *p*, and *d* wave functions of the form of an exponential times a combination of *r*, for a large number of different values of the exponential scale factors. The program also included instructions to the output unit so the results came out of the automatic typewriter in final table form, ready for offset photographic printing.

These tables will be published sometime this fall. I hope they will be useful to those people who haven't access to a machine like Whirlwind, but I am already sure we won't use them at Tech. For the calculation of such energy values is now an established subroutine, taking only a few seconds of machine time. It is much more efficient to ask the machine to recompute this value when it is needed than it is to look up the value, put it back in the machine and have it using up a storage register, waiting until it is needed.

Last fall Tubis programmed the machine to take the next step and carry out the variation to find the best values of the parameters. The general program specifies the nuclear charge and the electronic configuration desired; the subroutines developed earlier tell the machine how to compute a trial value of the energy for a given choice of parameters. Then the general program tells the machine to compute a sequence of energy values for a sequence of points in parameter space, how to direct this sequence so it approaches the minimal point, and, finally, to stop the sequence when the energy value has become stationary and to print out the corresponding energy and parameter values.

Another ambitious job, done last year by J. D. C. Little and F. J. Corbató, was the computation of a 500-page table of spheroidal wave functions. These wave functions can be expressed as a series of spherical harmonics, the successive coefficients in the series being related by a three-term recursion formula. The equation for the separation constant thus involves a continued fraction, which can be arranged to be solved by iteration. The program first had the machine perform this iteration until successive results agreed to eight decimal places; next, from the recursion formula, it was told to compute the ratios between the successive series coefficients; then, by applying a normalizing condition, to compute the coefficients themselves; and finally to arrange the results on the output tape in the proper order and with the proper instructions so that the automatic typewriter would title the page, give the values of the eccentricity parameter, the separation constants and the series coefficients in appropriately labelled columns, and would even print in dots so a draughtsman could draw lines separating rows and columns. This line ruling was the only hand work done on any of the 500 pages. They went to the photo-offset machine practically untouched by human hands.

This program also was done by persons with no previous computer experience. It took ten man-months for Little and Corbató to get an error-free program—there were about 2000 instructions in the final code. It took Whirlwind ten hours to do all the calculations and it took the automatic typewriter more than 200 hours to type the 500 pages of tables. Here again I hope the tables (they are published now [1]) will be of use to those not sitting next to a high-speed computer, but I don't believe they'll be used at MIT. For this program, stripped of the instructions for typing out the table, is now a part of our subroutine library; it only takes Whirlwind a fraction of a minute to compute a spheroidal wave function, for any order or any ratio of wave length to interfocal distance, when it is needed in a more general calculation.

For example, spheroidal wave functions are needed in the calculation of nuclear energy levels of spheroidal nuclei. This last fall the energy levels of a spheroidal, square-well potential were calculated on Whirlwind by J. L. Utetsky for different eccentricities. These calculations use the previously developed program to compute a sequence of wave functions, each coming closer to satisfying the boundary conditions, until some predetermined criterion of closeness of fit is satisfied and the corresponding energy is then recorded and a new sequence is started to close in on the next level. This more general program was fairly easy to devise because the most difficult part of it, that for the calculation of the spheroidal functions, was already done.

I COULD give several hundred similar examples, each with its lesson learned. But if I go on, I might be suspected of doing what I didn't want to do, of boasting about the capabilities of digital computers. Perhaps I have given enough examples to illustrate my earlier generalities: that high-speed computing machines don't do your thinking for you, that you have to work with them to get the most out of them, and that they aren't good for every kind of problem. Perhaps you have come to the same conclusion we have, however, that in spite of their shortcomings they are nice things to have around.

Nowadays it doesn't seem you can get very far in experimental physics without using big, expensive apparatus. You can't compete unless your machine costs a million or so. Well, we theorists are now in the running too; we needn't apologize for our cheapness and simplicity of operation any more. High-speed computers are our million dollar gadgets.

[1] Stratton, Morse, Chu, and Corbató, *Spheroidal Wave Functions*, John Wiley & Sons, N. Y. and Technology Press, MIT, 1956.