MATSCIENCE REPORT No. 67

PROCEEDINGS OF THE ONE DAY SYMPOSIUM

ON

COMPUTERS IN SCIENCE AND TECHNOLOGY

Edited by
K. SRINIVASA RAO

THE INSTITUTE OF MATHEMATICAL SCIENCES, MADRAS-20 (INDIA)

MATSCIENCE REPORT 67.

THE INSTITUTE OF MATHEMATICAL SCIENCES
MADRAS-20 (INDIA)

PROCEEDINGS OF ONE-DAY SYMPOSIUM ON COMPUTERS IN SCIENCE AND INDUSTRY[+]

Edited by

K.Srinivasa Rao,
MATSCIENCE, Madras.

[+]Held on 28th March 1969 at the Institute of Mathematical Sciences, Madras.

PREFACE

The one day symposium conducted on 28th March, 1969, by the Institute of Mathematical Sciences, Madras, was inaugurated by Professor Alladi Ramakrishnan, Director. In his inaugural address he pointed out that one can neither over emphasize nor under estimate the role played by modern computers in a developing country. Professor A.P. Jambulingam, Principal, Technical Teacher's Training Institute, was the Chairman of the morning session. Dr. S.Natarajan, Fundamental Engineering Research Establishment, spoke about the Synthesis of Finite-State machines and pointed out the role of Boolean algebra in computer design. Dr. K.Ananthanarayanan, Matscience, emphasized the vital role of Computers in solving problems in Nuclear Physics. Dr. V.Devanathan, University of Madras, pointed out the differences between two of the programming languages, ALGOL and FORTRAN.

In the afternoon session, Mr.K.Srinivasa Rao, Matscience, highlighted the uses of Computers in Industry, Viz. Games, Printing and Editing, Machine translation of languages, design of Optical systems and Crystallography. Mr.M.S.Jayaraman, International Computers and Tabulators (India) Pvt. Ltd., dilated upon the thrilling idea of building up of a thinking machine and stressed in particular the uses of Computers in medical diagnosis and treatment of diseases. Mr.S.Nagarajan, Matscience, dealt with non-linear problems in Fluid Dynamics that can be attempted only with

Computers. Mr.S.Sivaraman, International Business Machines Corporation, spoke on the role of IBM in providing Computer facilities in India. He discussed the relative efficiencies and capacities of various IBM Computers now in Madras.

This report contains in addition to the articles of the lectures in the one-day symposium, a contributed article by Dr.K.Chandrasekharan and Mr.P.K.Seshan, of the Chemical Engineering Department, A.C.College of Technology, Madras.

K.S.R.

C O N T E N T S

A NUMERICAL INTEGRATION OF SCHRODINGER EQUATION

K. Ananthanarayanan,
Matscience, Madras.

To find the binding energy of two particles in s^2 configuration with a hard core two body potential, the exact solution of the wave function is needed, since the Born approximation does not work. Actually,

$$\int_0^\infty u^2(r)\, v(r)\, dr = \infty$$

and
$$[-\nabla^2 + x^2 + V(bx)]\,\psi(x) = \epsilon\,\psi(x) \qquad (1)$$

is the Schrodinger equation for $\frac{D}{b} = x_c \leq x < \infty$, where we have used $b = \sqrt{\frac{2\hbar}{m\omega}}$ as the unit of length and $\frac{1}{2}\hbar\omega$ as the unit of energy. The boundary conditions are

$$\psi(x_c) = \psi(x \to \infty) = 0 \qquad (2)$$

We now expand $\psi(x)$ in a set of orthogonal functions

$$\psi(x) = \sum_n a_n \varphi_n(x) \qquad (3)$$

where
$$\varphi_n(x) = \frac{R_{no}\left(\frac{x - x_c}{\lambda}\right)}{x} \qquad x_c \leq x < \infty \qquad (4)$$

$R_{no}(x)$ is the harmonic oscillator wave function for $l=0$,

$$R_{nl}(x) = N_{nl}\, x^{l+1}\, e^{-x^2/2}\, L_{n-1}^{l+\frac{1}{2}}(x^2) \qquad (5)$$

where
$$N_{nl} = \left(\frac{2(n-1)!}{[\Gamma(n+l+\tfrac{1}{2})]^3}\right)^{1/2}.$$

Here λ is a variational parameter which is used to scale the wave functions. Substituting (3) in (1) and using the harmonic oscillator differential equations

$$\sum_{n} a_n \Big\{ \Big[\frac{\epsilon_n^0}{\lambda^2} + x_c^2 - f \Big] \delta_{n'n} + \int_0^\infty dz\, R_{n'0}(z) \Big[\Big(\lambda^2 - \frac{1}{\lambda^2}\Big) z^2 + 2\lambda x_c z \Big] R_{n0}(z) + \int_0^\infty dz\, R_{n'0}(z)\, V[b(\lambda z + x_c)]\, R_{n0}(z) \Big\} = 0 \qquad (6)$$

where $\epsilon_n^0 = 4n-1$ is the oscillator energy. This is an exact infinite dimensional matrix eigen value equation. But it can be shown that by restricting the values of n and n' - i.e. by restricting the dimensionality of the matrix to be finite - we can solve eq.(6) to get the lowest eigen value of the restricted matrix which will be an upper bound for the true eigen value. For a restricted dimensionality of the matrix, we vary λ until a minimum is found. The size of the matrix is increased and the whole procedure is repeated. It is found that after n=7 any increase in the dimension of the matrix changes the eigen value only very slightly. So that we stop the computation around n=7. The evaluation of

$$\int_0^\infty dz\; R_{n'0}(z)\, V[b(\lambda z + x_c)]\, R_{n0}(z)$$

can be done in terms of Talmi integrals. The Talmi integral is defined by

$$I_p = \frac{2}{\Gamma(p+\frac{3}{2})} \int_0^\infty e^{-x^2} x^{2p+1} V(x)\, dx$$

Due to the rapid decrease of e^{-x^2} with x it was found that, for large p, the upper limit of the integration can be curtailed at x = 12, while for small values of p an upper limit of x=6 gave reliable results. Simpson's rule for numerical integration was first used. For n=7, we needed 13 integrations. As we take different values for the strength and range of the parameters and the various values of λ we had to do about thousand (3 x 5 x 8 x 13) integrals. The diagonalization of the matrix to find the eigen value ϵ was also done using the computer. The problem took just one hour of IBM7090 time.

For large values of p the function is peaked. So, the simple Simpson's rules fails to yield good results. A new integration routine was developed. This routine chooses more number of points near the peak and less number of points at other places.

Description of the routine:

As pointed out earlier, the usual integration subroutines do not work for peaked functions. In such cases, we have to use a routine which will take a large number of points in the region where the function is peaked and fewer points elsewhere. When the integrand is not known a priori the computer should find the position of the peak and do the integration properly. Both these processes can be combined with a single process as follows:- Let a and b be the lower and upper limits of the integral. Divide the interval (a,b) into two equal intervals. Taking the

value of the function at the end points and the middle point ($C = \frac{a+b}{2}$) we can find the approximate value of the integral using the Simpson rule:

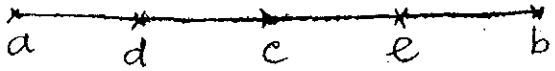
$$I_o = \frac{f(a) + 4f(c) + f(b)}{3} \times \frac{(b-a)}{2}$$

Now, divide the interval (a,b) into four equal intervals. Consider points

a,d,c on the one hand

and points c,e,b on the

ether hand to compute (using Simpson's rule):



$$I_1 = \frac{f(a) + 4f(d) + f(c)}{3} \frac{(b-a)}{4}$$

and $$I_2 = \frac{f(c) + 4f(e) + f(b)}{3} \frac{(b-a)}{4}$$

If the integrand is an extremely smooth function then I_o will not differ very much from the sum: $I_1 + I_2$. But, if the function is not smooth then the difference $I_o - (I_1 + I_2)$ will be large, compared to a preassigned quantity Δ then divide the interval (a,c) into four equal intervals (or (a,b) into 8 equal intervals) and compute

$$I_{11} = \frac{f(a) + 4f(k) + f(d)}{3} \times \frac{(b-a)}{8}$$

and $$I_{12} = \frac{f(d) + 4f(j) + f(c)}{3} \frac{(b-a)}{8}.$$

If the difference between I_1 and $(I_{11} + I_{12})$ is smaller than Δ stop the procedure, and use the same technique to the second half (c,b) of the integral and compare I_2 with $(I_{21} + I_{22})$ to find whether it is smaller than Δ or not. This procedure of splitting the intervals into twice the number of intervals is continued until each difference is less than Δ .

A program in FORTRAN IV language has been written by Ebbenyman (University of Lund, Lund, Sweden) using this procedure. This is given in the appendix.

APPENDIX

```
      SUBROUTINE INTEG (A,B,E,F,FI)
      EXTERNAL  F
      DIMENSION OR(23), AB(23)
      EPS = E
      LOGICAL DONE
      DONE = .FALSE.
      N = 3
      FI = 0.0
      AB(1) = A
      OR(1) = F(A)
      AB(2) = 0.5 * (A + B)
      OR(2) = F(AB(2))
      AB(3) = B
      OR(3) = F(B)
100   A1 = 0.5 * (AB(N) + AB(N-1))
      O1 = F(A1)
      A2 = 0.5 * (AB(N-1) + AB(N-2))
      O2 = F(A2)
      IF(DONE) GO TO  10
      FAV = 0.2 * (ABS(OR(1)) + ABS(O1) + ABS(OR(2)) + ABS(O2) +
                                                  + ABS(OR(3)))
      ERR = 45.0 * EPS * FAV
      IF (ERR.LT.1.E-25.OR.ABS(A-B).LT.1.E-15) GO TO 30
      DONE = .TRUE.
```

```
10 CF1 = 0.125 * (3.0 * OR(N) + 6.0*OR(N-1) - OR(N-2))
   CF2 = 0.125 *(3.0*OR(N-2) + 6.0*OR(N-1) - OR(N))
   IF (ABS(CF1-01 + CF2-02).GE.ERR) GO TO 20
   FI = FI + (AB(N)-AB(N-2))*OR(N) + 4.0 *(01+02) + OR(N-2) +
                                          + 2.0* OR(N-1)

   IF (N.EQ.3) GO TO 40
   N = N-2
   GO TO 100
20 IF (N.GT.21.OR.ABS(A2-A1)* 1.E4.LT.ABS(B-A)) GO TO 30
   AB(N+2) = AB(N)
   OR(N+2) = OR(N)
   AB(N) = AB(N-1)
   OR(N) = OR(N-1)
   AB(N+1) = A1
   OR(N+1) = 01
   AB(N-1) = A2
   OR(N-1) = 02
   N = N+2
   GO TO 100
30 PRINT 31
31 FORMAT (26H INTEGRATION NOT COMPLETED)
40 CONTINUE
   FI = FI/12.0
   PRINT 32, FI
32 FORMAT (5X, F 10.5)
   RETURN
   END
```

THE COMPUTER IN CHEMICAL ENGINEERING

K.Chandrasekharan and P.K. Seshan *
Department of Chemical Engineering,
University of Madras,
A.C. College of Technology, Madras-25.

***

The field of Chemical Engineering has been no exception to the influence of the computer. Chemical systems are unique in that their characterization, almost invariably, is associated with a number of variables(like temperature, pressure etc.). Hence chemical engineering design procedures, if they should be rigorous, will involve complex mathematical modelling and plenty of numerical calculations. Where simplification is attempted by linearizations, tedious step by step calculations are required for obtaining a reasonable degree of accuracy. In addition, where implementation on an industrial scale is involved, systems engineering approach and optimization of the process become essential. For all the above, it need hardly be stressed, that the aid of the computer is indispensable.

Computers are also used for 'on-line' control of chemical processes apart from the more routine application for data processing. Some of the reactions in chemical processes, even after very careful design, may develop operating characteristics (non-linear, stochastic etc.) which, unless closely watched and controlled to keep within limits, may lead to very explosive situations. In such

---

* Junior Research Fellow of U.G.C.

cases, an on-line computer would be very necessary.

Though for some of the applications mentioned above, the analog computer may be more suitable than the digital, in this article, we shall not go into a discussion of their relative merits, but shall restrict our interest to the digital type only.

We will now list here some of the important applications of the computer in the field of chemical engineering. They are:

1. To remove the drudgery of numerical calculations.

2. To remove the drudgery associated with 'trial and error' procedures of design which are standard in some of the unit operations such as heat transfer.

3. In checking mathematical models formulated to describe a process to help in engineering and development work.

4. In reactor dynamic analysis and process kinetics.

5. For 'simulation' of process. For example, continuous processes in refineries and petrochemical complexes can be simulated. This technique has proved to be a powerful tool in process development and also in 'expansion redesign' of existing plants.

5. In 'process control', where an important aspect of design is to produce a stable system. The computer can be used to calculate the responses expected of various types of disturbances by numerical techniques, irrespective of the complexity of the equations describing the dynamics of the system.

6. For 'optimization', by which we mean determination of the best possible way of doing a thing under a given set of conditions, the criterion of optimality being contained in the ''maximization' or 'minimization' of what is called the 'objective function'. Various techniques such as 'linear programming', 'dynamic programming' are in vogue.

7. For 'statistical computations' connected with analysis, correlation of experimental data, design of experiments and statistical quality control.

8. For 'evolutionary operation' which concerns with obtaining along with normal production in industry, information on 'how to improve production?'

While the applications listed above are specific to the field of chemical engineering, the details of problem analysis, mathematical formulation, computer orientation and programming are generally standard basic steps that are common to all engineering disciplines.

For illustration, we would take up here a problem in the unit operation of mass transfer (i.e, fixed-bed adsorption) in the field of chemical engineering and show how the computer can be put to use in helping to get the solution for the same.

ILLUSTRATION:

Fixed-bed Adsorption

Adsorption on an industrial (and hence pilot-plant) scale is usually carried out in a fixed-bed. A fixed-bed system is one in which the solid adsorbent is held stationary over a suitable false bottom in a vertical cylindrical column and the gas to be

purified of a solute impurity is passed through the bed at a convenient velocity for the adsorption to take place. The adsorption cycle is followed by a desorption step using hot inert gas for regeneration of the bed, before the next adsorption cycle is begun.

The effluent gas in an adsorption cycle, when analysed for the solute impurity will be found to give, on a time-scale, (volume of effluent under constant rate of flow) a graph of the type shown in the Figure 1, which is

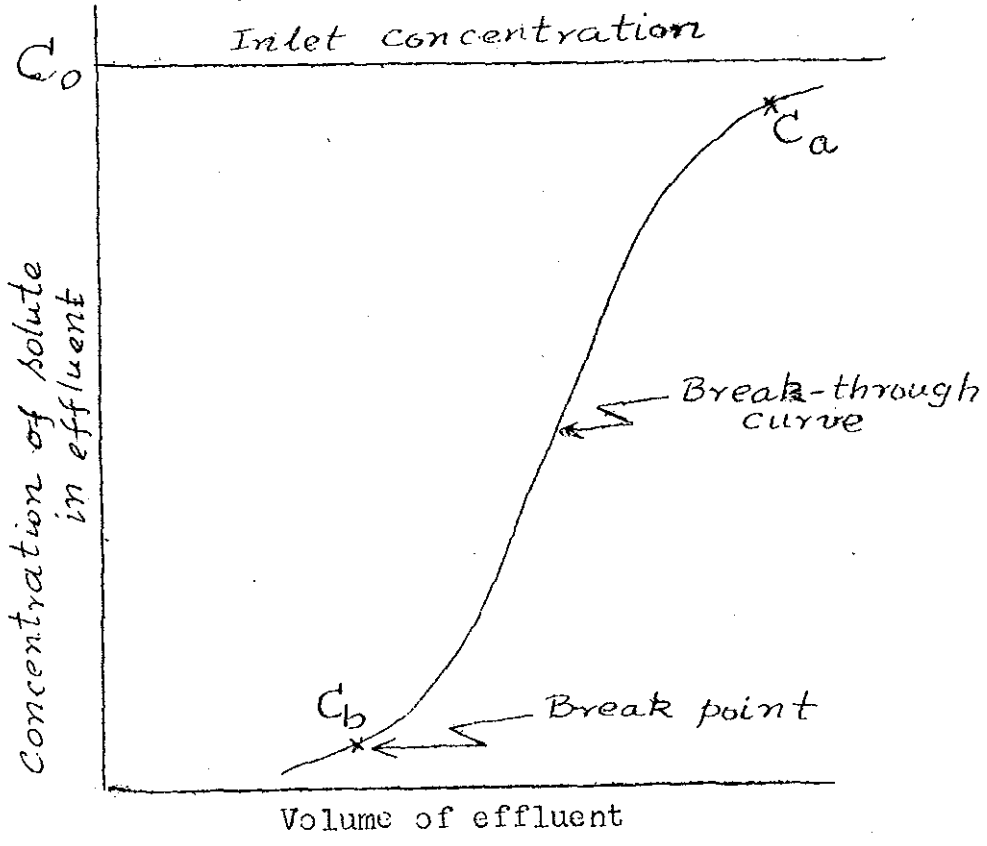


Fig.1. General concentration - history curve

called a 'breakthrough' curve and this represents the resultant effect of all the complicated processes that are taking place in the bed. Incidentally, the 'break point' which is the first occurrence of a rise in the effluent concentration since the entry of the gas into the column, is in itself an important design factor, as, in purification processes, it is the freedom of the effluent gas from even traces of the impurity that determines the adsorption cycle time.

Analysis

There are quite a few theoretical approaches developed to predict the breakthrough curve for a given set of conditions of the bed and the gas flow. Even to check up as to which theory is best applicable one need to have a few runs on a pilot plant fixed-bed adsorber. This would incidentally give a set of data which can be analysed with the aid of the digital computer, irrespective of the theories proposed, and with a minimum number of assumptions to aid computation. In fact, a programme can be made for the digital computer by which the best value of the solid-phase mass transfer coefficient can be obtained by direct numerical integration of the true non-linear mass balance equation defining the adsorber dynamics:

$$D_L \frac{\partial^2 C}{\partial u^2} - v \frac{\partial C}{\partial u} - \frac{\partial C}{\partial t} = \frac{1}{m} \frac{\partial q}{\partial t} \qquad (1)$$

The programme will also give the point mass transfer coefficient values at various levels of the bed, so that its variation, if any, may also be evident. Experimentally one would require only provision for tapping out samples of the gas phase at equal intervals along the height of the bed. The gas phase is allowed to enter the bed at time t = 0 and the inlet gas concentration of the solute is taken as a step function.

The main purpose of the set up will, therefore, be to make available data on the values of $C_{i,j}$ which is solute gas concentration on space, time coordinates.

For defining the equilibrium, Langmuir's adsorption isotherm is made use of in the form

$$q = \frac{k_2 C^*}{1 + k_1 C^*} \qquad (2)$$

The other equations made use of are:

the rate equation:

$$\frac{1}{m} \frac{\partial q}{\partial t} = u (C - C^*) \qquad (3)$$

and the velocity relationship

$$v = v_0 \frac{100 + C_0}{100 - C}$$

The Schedule:

The concentration values of the effluent gas stream form two dimensional array varying with grid numbers (or bed height from the upstream end) on one side and with the time index on the other.

The schedule of computations will involve:

a) Fitting by least squares technique a bivariate polynomial for the experimental values of concentrations $C = f(h, t)$ where h is any height of the adsorber column, t any time after the start of the experiment, and f the polynomial function) so that at any time and at any height the effluent concentration can be obtained from the polynomial.

b) Using the unsteady state non-linear material balance equation(1) to determine the partial derivatives of the variation of solute concentration in solid (q) with time in the various grid points and time intervals ($\partial q/\partial t$). In this calculation, a suitable value of the longitudinal diffusion coefficient D_L (which is usually small or negligible) may be used.

c) The values of ($\partial q/\partial t$) varying with time for each grid are integrated over time to determine the solute concentration q in solid at various instances for each grid.

d) Every value of q for each grid is substituted in the Langmuir's non-linear isotherm to obtain the surface gas concentration C^* corresponding to it.

e) The values of C^* are put into the rate equation to obtain the local mass transfer coefficients for every (h,t) considered, and, finally

f) An effective overall mass transfer coefficient U_o is calculated from the formula

$$U_o = \frac{\frac{1}{m} \sum_i \sum_j (\partial q/\partial t)_{i,j} (C_{i,j} - C^*_{i,j})}{\sum_i \sum_j (C_{i,j} - C^*_{i,j})^2}$$

Details of the numerical procedures are not elaborated here as it is out of scope of the objective of the present symposium.

ON PROGRAMMING LANGUAGES

V.Devanathan
Department of Nuclear Physics,
University of Madras,
Madras-25.

***

Electronic Digital Computers are capable of performing simple arithmetic operations such as addition, subtraction, multiplication and division but they do them with much greater speed and accuracy than would be possible by any other practical means of computation. Of course, they have the ability to make decisions on either/or basis and take alternative courses of action. For example, the machine could be instructed to take one course of action if a variable X was negative and another if it was positive or zero.

Digital computers cannot compete with their designers in the ability to solve complex problems. In fact, the machines themselves are not able to solve even the simplest problem without human help. The approach required to solve a problem expressed as a precise, step-by-step means of attaining the solution, must be provided to the computer. It is this set of step-by-step instructions that will be called a program. Programming a computer, then, is the development of such a set of instructions.

There are two steps in the writing of a program. One is the expression of the numerical process in a formal manner such as a flow charts. The second is the actual writing of the program in a language which the machine can accept.

In one sense, the computer program was created when the chart was developed. It fits most of the criteria since it is broken down into simple operations in a proper sequence leading to a solution. However, the computer is not capable of understanding the program in this form. Therefore, the flow chart must be transcribed into a form which the computer will accept.

Each computer has its own internal language. Usually, this language, while highly appropriate and meaningful to the given machine, is highly foreign to a human programmer. In fact it is called machine language.

To make communication easier for a human problem solver, languages have been developed which can be mechanically converted into machine language. These are called procedure oriented languages since they are ways in which well defined procedures (i.e. flow charts, etc.) can be expressed in a form palatable to the computer.

The machine itself effects the conversion of the program to its internal language through a program called a compiler. The compiler is itself a program written for the particular machine, usually by the manufacturer. A particular program which is to be translated (or compiled) is provided to the compiler as data. The result (or output) of the compilation is a machine language equivalent to the input program. This operation might be depicted as follows:

The input program is usually called the source program and the resulting machine language is called the object program.

The basic tenets of programming remain the same but computers manufactured by different firms use slightly different programming languages such as FORTRAN, ALGOL, COBAL and AUTOCODE. Of them, the first two are widely used for scientific purposes and we shall briefly compare the FORTRAN and the ALGOL and discuss their relative merits and defects.

In ALGOL, at the very outset, all the identifiers used in the program are declared. This declaration serves to provide the ALGOL computer with information about identifiers in the program. For example, a declaration is required to specify that the identifier for a particular variable is always to represent an integer. There are eight different declarations:

1. type declaration
2. Array declaration
3. label declaration
4. file declaration
5. format declaration
6. list declaration
7. diagonostic declaration
8. procedure declaration.

These eight declarations provide the compiler with all the information which it requires about a given program. This is in contrast to the FORTRAN, wherein the declarations of the identifiers are not required.

ALGOL is more elaborate but less confusing to a beginner. The FORTRAN arithmetic statement

$$x = x + 1.0$$

is a mathematical absurdity. But in FORTRAN language, what is meant is that X+1.0 is replaced by X by changing its value. In ALGOL, the equivalent statement reads

$$X \leftarrow X + 1.0$$

whereby the meaning is made transparent.

In FORTRAN, variables are of two types: fixed point and floating point variables. In the earlier version of FORTRAN, these two modes cannot be mixed in a statement. Of course, one mode can be converted into another by a separate statement. This is very highly restrictive and in the later versions of FORTRAN, the mixing of the modes is allowed. In the ALGOL, of course, there is no such distinction and the variables, real and integers, can be freely mixed.

In ALGOL, the subscripted variables are easily distinguished from the Function subprograms (Real Procedure). For subscripted variable (array), square brackets are used whereas for Real Procedure, round brackets are used.

Subscripted variable	CL[M,N]
Real Procedure	CL(M,N)

But in FORTRAN, a confusion arises between these two since the round brackets are used for both. If no function subprogram is there for CL, then it has to be understood as an array.

In ALGOL, the upper and the lower bounds of arrays can be adjusted. For instance, if A is a one-dimensional

array of 21 elements, one can write

A [0; 20] or A [-10 : 10]
or A [1 : 21]

Of course, for the sake of efficiency, it is recommended that the lower bound is kept as 0. But such a facility of adjusting the lower bound is not available in FORTRAN, wherein the lower bourd is always kept as 1. This causes very great difficulty and one has to resort often to auxiliary statements to circumvent this defect. For example, if one wishes to represent the factorial of integers as an array, then one finds it difficult to represent the factorial of zero. Factorial of zero is one but zero is not permitted as a subscript in FORTRAN !

Another advantage with ALGOL is that an array of any dimension is permitted but in FORTRAN, the dimension of the array is restricted to three.

In ALGOL, the following "DO STATEMENTS" are allowed.

1) FOR N ← 2 STEP 0.01 UNTIL 3.1001 DO

2) FOR N ← 1 STEP -2 UNTIL -11 DO

3) FOR N ← 1, 2.5, 3.5, 4.0, 4.25 DO

But in FORTRAN, the "DO STATEMENTS" are restricted to integer increments and only to equal increments.

The above discussion on ALGOL and FORTRAN is not complete but only selective. Each has its own restriction and effects are being made to remove these restrictions and improve the applicability. For example, in FORTRAN IV, complex quantities are permitted.

TOWARDS BUILDING UP OF A THINKING MACHINE

A BRIEF INTRODUCTION TO THE DEVELOPING SCIENCE OF CYBERNETICS

Mr.M.S.Jayaraman,
International Computers (India)Ltd., Madras.

The science of Cybernetics developed originally from a comparative study of machines and living organisms. As and when the need arose, the human brain always invented self regulating mechanical systems, all the time unaware of the fact that, the machines that were being made were a remote analogue of the human brain itself in its operations and functions. A realisation of this particular fact lead to a possibility of a comparative study of machines and living organisms, which would enable man to invent better machines and also acquire a better knowledge as to how living organisms actually function.

It is very difficult to give a clear-cut definition of cybernetics, since this field of activity embraces several branches of science; like electronics, mathematics, biology, medicine and communication theory. This is a field where a physiologist tells the engineer as to how better machines can be built and the engineer tells the physiologist as to how life functions. Naturally one question that a cybernetician has to face from a lay man is "Can the machines even start "thinking"?"

A comparative study of the human brain and an Electronic Computer may help us to search for an answer to the question of building "thinking" machines. The human brain is the most

remarkable control mechanism ever known to Scientists. It is very small in size, weighs hardly 3 pounds, consumes not even 20 watts of power and has a memory equivalent to a million million 'bits' of information. It learns from experience tremendously fast and can recall even information registered decades ago.

But, compared to a computer, it is just too slow in processing information, often forgets important things and succumbs to fatigue and malnutrition. It ages along with the man and becomes weak in all respects as the pathological changes take place with passing time.

The attempts of the engineer who designs the computer is always towards making good these short comings of the human brain, by making a machine which will perform those functions much better than the brain itself. In this respect, the Machine has scored a big march ahead with regard to speed. Computers can perform jobs in seconds which a human brain cannot perform in many hours or days.

In the field of biology efforts are being made at several laboratories to design artificial heart, lung, kidney and every possible element of the human body. The electronic computer is being made use of for diagnosis and also treatment of the diseases. In an experimental competition between a computer and a doctor it was noted that though the results of the

diagnosis were the same, the computer was able to point out certain very rare diseases, which the doctor could not find out. After all the doctor is human, and the human memory forgets things which do not occur frequently. Also the computer processing is unbiased and there are no moments of 'fatigue' for the electronic computer when its observations can go wrong.

But inspite of all these special merits, the stage has not yet been reached where the computer can replace the doctor completely. The symptoms of diseases are numerous. There are different layers of importance with regard to the symptoms. The human brain displays a high degree of flexibility in comparing and evaluating the symptoms of diseases and draw the final conclusion from a vast array of factors. The machines have not come up to this level - in fact they are still far behind. As things stand, the diagnosing computer machines, play a secondary role as the doctor's assistant. It is the doctor's word which is final.

The assistance of computers in medical diagnosis can be illustrated by one simple instance. The analysis of an electrocardiogram graph is a method of finding out a heart ailment. Here the doctor will find it difficult to analyse micro variations, which have been smoothened out by the formation of the curve. But, when a computer analyses a cardio gram graph it resolves the components of the curve into very small components and can detect and point out very minor variations which

do not strike the human eye. In fact, the computer can point out the possibility of an apponently health person acquiring a heart ailment within a short period. Such medical forcasts are impossible, without a machine.

A cybernetician always finds similarities between machines and the human organisms; studies the human organism with the help of the machine and simultaneously improves the machine by observing how the human organism functions. As an author points out, a cybernetician driving an automobile can think of a 'rheumatic' ailment affecting his vehicle whenever the engine squeaks pitiably. If the corburator wheezes while stepping on the gas, he may exclaim that "it is suffering from Asthma!"

If we compare the fundamental aspects of machine operations and body functions, we find that they are basically same in nature. The nervous system of the human body can be compared with the mechanical control systems like the automatic regulators, governors etc. which are parts of mechanical control systems. If one carries out a process of abstraction, it will be found that the operations of all these mechanisms can be described in similar mathematical terms. The terms or formulae can be same whether you describe or service a steam engine, an automatic steering device or a nervous system.

It is this similarity between man and machine that is prompting man towards building up a 'human-machine'. In the field of Bionics, a study is being made of the electronic neuron analogues and their utility in computing machines. Work is now being carried out on machines capable of interpreting speech, writing and even visual images directly.

Time was when the words 'thinking machine' were just a mockery. Later, with the development of computers and its influence on several fields of activity, man went to the other extreme and started thinking that the machine can do just anything that the human can. Most of the 'computer-jokes' will testify this point of view. But, nowadays, an informed lay-man knows the merits and also the limitations of the machines.

As for man's ability to build 'thinking' machines there is a wide variety of opinions, even amongst cyberneticians themselves. Some feel that the time is not far off. Some others say that a machine can replace man only when it is built in with 'consciousness'. Some animals can 'think' and no animal is capable of conscious action which requires the development of a definite attitude towards things and not a mere passive awareness of the environments. What we want at the final stage is to have a machine which can possess a definite 'attitude' towards things i.e. an artificial creature with 'psychological' qualities, and the 'human ego'. Can man reach such a stage? Time alone can answer this question.

NONLINEAR PROBLEMS IN HYDRODYNAMICS

S.Nagarajan,
MATSCIENCE, The Institute of Mathematical Sciences,
Madras-20, India.

In this symposium so far interest has been concentrated on such applications of computing which makes life easier in many a calculation of engineering, physics and allied sciences. We shall here illustrate a similar adventure - in computing, which is doing in spirit - in that it tried to explore such regions of mathematics and physics, where there is a paucity of even basic analytic information about the mathematical procedures. Since our audience here comprises of people who have a common link of interest in computing, one shall discuss only numerical and digital niceties here and leave the implications of the results in the relevent fields of application to a different report.

Construction of numerical procedures for solution of Linear Differential Equations, for example as in the Schroedinger equation with a prescribed potential proceeds very often from a known analytic solution, through an approximation generally classified under either a Born approximation sequence or a perturbation sequence. Thus there exist fairly well-defined techniques for eigenvalue problems, which are either matrix inversion or variational procedures. But for partial differential equations which are generally either Elliptic or parabolic, already one hits a snag - in that the numerical procedure which very often involves an integration in space or time from a known initial or boundary value is much more cumbersome and elaborate. Further

if the equations in addition are non-linear or integro-differential in character - the computing philosophy gets blurred indeed. We shall in this lecture try to look at one such explicit application - with an idea of illustrating with a difference the advantages of non-linearity rather than the disadvantages. We shall give two examples, one that leads to the generalisation of the Runge-Kutta procedure to a set of non-linear integro differential equations and another that involves an iterative solution to a set of non-linear integral equations. The relevent physical situations, wherein these arise have been discussed in great detail elsewhere (Nagarajan and Kraichnan (1967), Nagarajan (1969a), Nagarajan (1969b))

1. RUNGE-KUTTA PROCEDURE FOR NONLINEAR INTEGRODIFFERENTIAL EQUATIONS

We shall concern ourselves with a particular class of equations which can be written in the form

$$\left[\frac{\partial}{\partial t}+\nu_\alpha(k)\right]\psi_\alpha(k;t) = \iint dp\,dq \int_{-\infty}^{t} dt' \qquad \Delta \equiv (\underline{k}=\underline{p}+\underline{q})$$

$$\left[A_{\alpha\beta\gamma}(k,p,q)\,G_\alpha(k,t-t')\,\psi_\beta(p,t')\,\psi_\gamma(q;t') - B_{\alpha\beta\gamma}(\underline{k},\underline{p},\underline{q})\,\psi_\alpha(k,t')\,G_\beta(p,t-t')\,\psi_\gamma(q;t-t')\right]$$

$$\left[\frac{\partial}{\partial t}+\nu_\alpha(k)\right]G_\alpha(k,t) = \iint_{\Delta} dp\,dq \int_{0}^{t} dt'\, C_{\alpha\beta\gamma}(k,p,q)$$

$$G_\alpha(k,t-t')\,G_\beta(p,t-t')\,\psi_\gamma(q,t')$$

These equations are to be solved forward in time given the values of $\Psi_\alpha(k)$ at $t = t_o$ and the condition that $G_\alpha(k,0) = 1$ and $G_\alpha(k,t) = 0$ for $t < 0$. These equations are convolutory in character in and integro-differential in time. The suffix denotes that one have a set of equations or that the and G are matrices in general. The convolutory character of the p,q integrals poses a problem in that the p-q integrals are not indepndent and that we have to constrain our integration - in such a way that the condition $\underline{k} = \underline{p} + \underline{q}$ is not in our digital procedures for discretisation of the infinite k-range. In other words as we integrate forward in time, at each time step, we have to perform the convolution. Economy in representation memory requirement and integration procedure (for the convolution) thus is not academic but crucial.

Thus for a typical convolution term

$$\int\int_{\triangle} dp\, dq\; R(k,p,q) \longrightarrow \sum_p \sum_q V_{kpq} R(k,p,q)$$

where V_{kpq} are precalculated weight factors which take care of the pq restriction and are such that $V_{kpq} = 0$ for $p < q$. This reduces the range of integration each time. Further from a digital point of view, it facilitated reduction of the range of Do-loops. These V-factors are calculated once and for all for a variety of sample functions R(k,p,q) and are fed with the machine - at the start. The difficult calculation of these weight factors will be appreciated if one notices that the discretisation

of the must be fine enough to take care of the restriction faithfully for the entire discrete region of k,p,q. If we discretise k such that the entire range of k is divided into n-steps such that the ranges of k are specified by their medium points and the upper and lower bounds of subsequent intervals coincide.

$$k_{lower}(i) = k_{upper}(i-1)$$

$$k(i) = k(i-1) + \Delta k(i)$$

$$\Delta k(i) = k_{upper}(i) - k_{lower}(i)$$

The interval $\Delta(k(i))$ can be chosen either constant or logarithmic. We calculate the allowed volume-elements in the integral $\sum\sum_{k=p+q}$ for the various classes of triangles in which either p,q or both are large or small compared to k or of the order of k. These characterisation of the triangles depend only on the relative magnitudes of rather than on their absolute magnitudes. Thus one can see for a given choice of $k(i)$, $p=k(j)$, $q=k(l)$, the triangle condition is satisfied only for a region of values of k,p,q, in the range Δk , Δp , Δq and in particular the character of the triangle would change in the range. Thus we can evaluate the various V factors through a detailed trigonometric calculation.

This finishes the preliminary phase of the calculation. The second part is the inclusion of a suitable integration scheme - for the differential part.

To do this we look at the integrodifferential equation in this fashion

$$\frac{d}{dt}\psi_\alpha(k,t) = -\nu_\alpha(k)\,\psi_\alpha(k,t) + \int_{t_0}^{t} dt' f_\alpha(k,t,t')$$

$$\frac{d}{dt}G_\alpha(k,t) = -\nu_\alpha(k)\,G_\alpha(k,t) + \int_0^t dt'\,\theta_\alpha(k,t';t)$$

where

$$f_\alpha(k,t',t) = \iint_\Delta dp\,dq\,\Big[A_{\alpha\beta\gamma}(k,p,q)G_\alpha(k,t-t')\psi_\alpha(p,t')\psi_\alpha(q,t') - B_{\alpha\beta\gamma}(k,p,q)\psi_\alpha(k,t')G_\beta(p,t-t')\psi_\gamma(q,t-t')\Big]$$

$$\theta_\alpha(k,t',t) = \iint dp\,dq\,\Big[C_{\alpha\beta\gamma}(k,p,q)G_\alpha(k,t-t')G_\beta(p,t-t')\psi_\gamma(q,t')\Big]$$

We discretize $\iint dp\,dq \to \sum_i \sum_j V_{k\,pq}(i,j)$

$$\int dt' \to \sum_i \Delta t(i)\,, \quad t(0) = t_0$$

We will have to resort two schemes of time-reckoning. For doing the time integral in each time step which depends on the past, one decide in advance that we will approximate the time domain by a set of steps. This will be necessarily small - for computational reasons. Within the first step, for example the integral trivially reduces to a sum of one term and

within this step the equation is integrated from t_0 to $t_0 + \delta T$ where δT is the time step corresponding to each of the Time-steps, using a variable step 4th order RUNGE-KUTTA SCHEME. The intermediate values of the ψ's and G's are not retained but only $\psi_\alpha(t_0)$, $G_\alpha(t_0)$ and $G_\alpha(t_0+\delta T)$, $\psi_\alpha(t_0+\delta T)$ From $G_\alpha(t_0+\delta T)$, $\psi_\alpha(t_0+\delta T)$ again the equation is integrated forward in time - with the proviso that at each small time step now, there are two terms to be added - to denote the time interval.

The whole scheme can be carried forward without much ado this way. But the complication increases geometrically for each addition of a time step. We use all possible auxiliary storage capacity of the machine, tape, disc etc - to facilitate this. The results of a similar calculation are given elsewhere (Nagarajan 1969b).

2. ITERATIVE PROCEDURES FOR NONLINEAR EQUATIONS

The procedure that we gave above is not very particularly dependent on non-linearity. It involved originality in treating convolutory behaviour and integro-differential structure only. We will now illustrate another example, where non-linearity plays an important role - in the solution.

We have a set of equations of the form

$$A_\alpha(k) = \frac{F^\alpha(k) + \sum_{\beta\gamma}\sum_{\substack{pq \\ \triangle}} R^{kpq}_{\alpha\beta\gamma}\left[A(\,),\zeta(\,),\eta(\,)\right]}{\nu_\alpha(k) + \sum_{\beta\gamma}\sum_{\substack{pq \\ \triangle}} S^{kpq}_{\alpha\beta\gamma}\left[A(\,),\zeta(\,),\eta(\,)\right]}$$

$$\eta_\alpha(k) = \nu_\alpha(k) + \sum_{\beta\gamma}\sum_{\substack{pq \\ \triangle}} P^{kpq}_{\alpha\beta\gamma}\left[A(\,),\zeta(\,),\eta(\,)\right]$$

$$\left[\zeta_\alpha(k)\right]^{-1} = \left[\eta_\alpha(k)\right]^{-2}\left[\sum_{\beta\gamma}\sum_{\substack{pq \\ \triangle}} Q^{kpq}_{\alpha,\beta,\gamma}\left[A(\,),\zeta(\,),\eta(\,)\right]\right]$$

where P [ ] , Q [ ] , R [ ] and S [ ] are known non-linear functions of their arguments. And $F_\alpha(k)$ is given. The problem is to find A, ζ and η as functions of k. We start the calculation with a set of trial values of [A( ), ( ), ( )] and evaluate the functions P [ ] Q [ ] , R [ ] and S [ ] for a given choice of k,p and . We own through the set of values of p, which are allowed by the $\triangle$ restriction as in the first example and evaluate a new set of A( ), ( ), ( ) . We now construct a new set of (A, ζ , η) such that

$$A_{N+1} = \alpha_A A_{N-1} + (1-\alpha_A) A_N \;;\; \eta_{N+1} = \alpha_\eta \eta_{N-1} + (1-\alpha_\eta)\eta_N$$
$$\zeta_{N+1} = \alpha_\zeta \zeta_{N-1} + (1-\alpha_\zeta)\zeta_N$$

With this new set we make a fresh calculation and keep iterating till we get to a converged set such that $\left|A_{m+1}-A_m\right| / A_m$ $<< \delta$ a preassigned value for accuracy. This iteration converges much faster than it would in the linear case, since if we overestimate A, η or ζ at any stage, the next iteration underestimates it and so on.

A detailed account of such a calculation is given elsewhere (Nagarajan 1969a). Details of the procedure are given there.

REFERENCES

1) Nagarajan and Kraichnan, Phys. Fl. Vol.10, 859 (1967).

2) Nagarajan, Matscience Preprint No.25 - 1969 (1969a)

3) Nagarajan, Matscience Preprint, under preparation (1969b)

IBM'S ROLE IN SCIENTIFIC COMPUTING FACILITIES IN INDIA

S. Sivaraman
Systems Engineer,
International Business Machines Corporation,
Madras-2.

***

Computer population in India is increasing at a fast pace. We have today a variety of Systems and within each system a variety of configurations installed and on order. With growing use of computers, the demand on application development, software support, sophisticated application areas is also increasing. Customers are interested in system configurations for Remote Access Computing, Multiprogramming, Time Sharing etc.

IBM has been marketing its products in India since 1955 and over these last 14 years our personnel have gained lot of experience. In order that this accummulative experience could be of benefit to our customers on all India basis, IBM early in 1968 decided to promote Industry Marketing activity in selected key industries. These key industries were selected on the basis of IBM equipment either installed or on order like - Textiles, Steel, Insurance, Manufacturing, Railways and Education and Research. Industry Managers for these selected segments of our business were appointed. These industry Managers coordinate the marketing activity in their respective industries. Benefit of experience gained in one installation; for example, can now be passed on to other installations in the country. It avoids duplication of effort in areas where considerable effort has already been put in by either the customer or IBM- like development of software packages.

In university environment this is particularly important as Research work using computers is very active in India. Exchange of program packages for solving complex mathematical problems developed at one centre sometimes result in good saving of time. In IBM this exchange is not limited to only India but through contacts with respective Industry Managers in other countries, specific information like configuration, benchmark timing, program packages can be obtained.

The first scientific computer introduced by IBM in India is the popular 1620. We have nearly 15 of them installed in various parts of the country, with core capacity ranging from 20K to 40K and access time from 20 μs to 10 μs.

Today, IBM has introduced the latest third generation scientific computers the 1130 computing system and the system/360 Model 44.

I wish to give you the highlights of these systems.

The 1130 is an advanced stored-program computing system designed for use in engineering, research management science and business application through a wide range of system configuration. New solid-logic technology provide powerful processing capabilities in a compact low cost system. Stored program control by a flexible comprehensive instruction set including Boolean logic capabilities and double precision arithmetic. Fast access core storage ranges from 4K to 32K words of 16 bits . Memory speed ranges from 2.2 to 3.6 μs . Model II processor contains direct access disk storage with capacity for storing upto 512000 sixteen bit words on-line. Automatic hardware interrup system provides program-controlled meritoring

of overlapping input/output. Overlapped high-speed data transfer to core storage from direct access dish storage at 35000 words per second is accomplished on cycle stealing basis concurrently with computing operation. Input/output units can be chosen from, Card Read punches, high speed printers Visual Disply Unit, X-Y Plotter etc. Programming support includes an extensive library of scientific programme packages such as COGO, Mathematical subroutines, Numerical surface Technique and Contour Map Plotting, Statistical Systems etc.

Gentlemen, we have a 1130 System installed right across the road, at the Centre for Advanced Study in Physics of the University of Madras, located in the A.C. College of Technology Campus.

System /360 Model 44 is the member of the system /360 family, oriented specifically toward scientific applications.

During the last decade, computer applications have become more sophisticated than ever. Commercial applications have been made more scientific through the use of mathematical techniques like linear programming, PERT, CPM, etc. This has necessitated the development of faster processing units with large memories. Similarly, scientific applications have taken commercial tinge, requiring faster Input/Output units and editing facilities for neat printed output.

IBM realised this need and in 1964, came up with the universal computer system/ 360- covering commercial, scientific and teleprocessing requirements.

As I mentioned earlier, system /360 Model 44 is the member of the system /360 family oriented specifically toward scientific applications. Within the framework of system /360 architecture the model 44 emphasises fast parallel library operations, and short, long

and variable length precision arithmetic, with wide range of input/output equiment . The Model 44 may be interconnected to system /360 models 30 to 75 via their channel to channel adapter and to IBM 1800 Data Acquisition and Control Systems via its System/ 360 adapter. Similarly the IBM 1130 computing system can be hooked onto a system /360 Model 44. This design has produced a low-cost, high-performance system tailored to the application areas of general scientific computation, data acquisition and data reduction.

Some of the highlights of the system/360 Model 44 are

1. 32 bit word plus parity bit.
2. One μs per word memory cycle speed.
3. Full 32-bit-word parallel arithmetic and data paths.
4. Upto 256K bytes of memory positions.
5. Integrated single disk storage drive.
6. Short and long precision, Floating Point arithmetic feature including variable length long precision.
7. 16 general purpose registers with optional high speed implementation of these registers.
8. High Resolution Timer.
9. Low and High Speed Channels.
10. A wide range of I/O units.
11. Comprehensive programming support.

The model 44 utilizes the same advanced logic technology circuiting as the other models of system/360. The miniaturised circuit modules result in

1. Faster circuitry because of high-density packaging
2. Lower cost and greater design flexibility, made possible by simplified batch fabrication processes.
3. Less maintenance because of ruggedised components.
4. Easier maintainability because of increased checking.
5. Improved physical access to circuit elements.

Model 44 is designed to work under a programming system known as the programming system/44. It provides a fully monitored operating environment and supports FORTRAN IV and Assembler languages. With the addition of extra Disk Drives and the commercial Feature, the system can work under the Disk Operating System or the most powerful operating System.

In such a case, further languages like COBOL, PL/1, and RPG will also be supported.

Scientific Subroutine package, an example of the application packages provided by IBM, is a collection of over 250 FORTRAN subroutines divided into three groups; Statistics, Matrix manipulation and mathematics. They are a collection of input/output free, computational building blocks that can be combined with a user's input, output or computational routines to meet his individual needs.

SOME USES OF COMPUTERS

K.Srinivasa Rao,
MATSCIENCE, The Institute of Mathematical Sciences,
Madras-20. India.

"The past is but the beginning of a beginning,
and all that is or has been is but the twilight
of the dawn."

- H.G.Wells.

In recent years, it has become increasingly apparent that "any procedure which can be precisely described can be programmed to be performed by a computer". There has been a phenomenal growth in the number of computers - it has swelled from 10 or less in 1950 to 3,000 and more in 1960 - and there is practically no branch of Science or Industry which has not been benefited by the advent of modern high-speed digital computers. Though computers are only virtually infallible, they have undoubtedly enabled man to extend widely his horizon of achievements.

During the short time allotted to me, I shall mention a few fields wherein computers are used. I will attempt only to indicate how problems are computer oriented.

GAMES:

We are all aware that in January, 1969, Cricket history was created when a Johannesburg newspaper organised Computer Cricket Tests between players who probably would have met in

Test matches, if the M.C.C. tour of South Africa had not been called off. The first theoretical Test match ended in a draw when rain stopped play half an hour before the scheduled close. The next two tests were won, one each by M.C.C. and South Africa.

Programming computers to play Games is a convenient way of understanding the methods which must be employed for the machine simulation of human intellectual behaviour. Of all the games, chess still has the reputation of being considered as the intellectual game par excellence and so far it has defied all efforts at complete solution. This is because, the number of all possible positions of the Chess pieces is of the order of 10^{43}, which is a formidable number. Hence, the problem is not that of programming a machine to play perfect chess, which is quite impractical, or to play merely legal chess, which is trivial. But, the problem is to make the Computer play skilful chess, comparable to that of an intelligent human player. The first complete chess program was written for the IBM 704 computer by A.Bernstein and his collaborators in 1958. This program requires about 8 minutes to make a move, it looks four half-moves in advance and it plays a respectable and none-too-obvious a game of chess. The complexity of the problem is reduced by their program which selects for analysis seven most plausible moves out of the 30 or so possible moves. This selection is made on the basis of 8 questions which the program explores in order. Quoting Bernstein these questions are the following:

"1. Am I in check, and, if so, can I capture the checking piece, interpose a piece, or move away?

2. Are any exchanges possible and, if so, can I gain material advantage by entering upon the exchange, or should I move my man away?

3. If I have not castled, can I do so now?

4. Can I develop a minor piece?

5. Can I occupy an open file?

6. Do I have any men that I can put on the critical squares created by pawn chains?

7. Can I make a pawn move?

8. Can I make a piece move?"

It is to be observed that the choice of questions and their order follows the common lore of the chess world. A serious defect of this program is its possibility of overlooking simple moves which may later have very important consequences. Several other groups in USA, Europe and USSR are working in this field to overcome such drawbacks. One such attempt is that of developing special Information Processing languages.

A limited number of workers have programmed computers to play checkers or droughts, bridge, tic-tac-toe, nim and various business and so-called war games.

It is now generally accepted that computers can outperform man at certain simple games, especially if their human opponent is not clever at remembering the specific rules of the game and the various resulting possibilities. Certainly a human being feels a bit awkward after having been beaten by a machine in games like nim or tic-tac-toe but at chess the outcomes are still debatable.

Printing and Editing:

The printing and editing industry, was in the early sixties, more prepared than some other industries for the introduction of the computer into production process. For, even ten or more years before the advent of the computers, the employees were accustomed to paper tape as input to line-casting machines. Two of the most important aspects of automating printing are: Justification and Hyphenation.

In printing, equal and even spacing between words is the goal, but it is not necessary to have exact spacing between the words, for it is sufficient if they appear to be equally spaced. Insufficient spacing causes words to "run together" and too generous spacing causes "rivers" of white on the page - both reduce the legibility for reading. Justification is the process of making a line tight in the column width being used. Justification in machine systems is ordinarily performed by an operator who increases or decreases spaces between words, sometimes inserting thin spaces between the letters of the longest word and also hyphenates words whenever necessary. Programming a computer to adjust spacing has been found to be easier than programming a computer to hyphenate words. For, hyphenation points in English have been determined more by literary tradition than by logic; there are mnay exceptions to the "rules" which do exist. It is very difficult to program for all the exceptions in a language which probably contains 50,000 or 60,000 words.

Then, too, dictionaries used as "standards" often differ among themselves. All in all, writing a hyphenation program has been found to be extremely difficult.

One of the approaches to the problem of hyphenation is called the dictionary look-up program. In this approach, lists of some 25,000 words with all possible hyphenations indicated, are kept on duplicate reels of magnetic tape. Whenever the necessity for hyphenation occurs, the computer which can read forward or backward at 10,000 characters per second, searches the magnetic reels for the hyphenation of the word. Such a dictionary must be updated frequently. In 1964, a "sin" committed by a not upto-date hyphenation routine was to hyphenate a previously infrequently encountered name which had become swiftly important on the American scene as "GOL-DWATER".

When the computer takes care of the justification and hyphenation, not only are the clerks relieved by these botherations while typesetting but due to the elimination of loose and tight lines by the Computer programmes, the Computer brings out an output which is more uniform in appearance.

The North Carolina Computation Centre developed a program for Univac 1105, a large general purpose machine, to edit stories. All stories for an entire day form the input for the computer at a time. As each story enters, it is converted to its corresponding computer printer code. These are stored on a magnetic tape which is printed on an off-line high-speed printer. Working

with the print out, the editor selects the stories he wishes to use, writes headlines for them, indicates lines which he wishes to cut, prepares insertions and writes directions for merging stories. His instructions are punched on cards in predetermined Formats. These instructions are carried out by the computer which brings out the final form in which a page contains two 10-inch columns of type, in capital letters. In 1963, Computer programs for editing news items were also attempted.

Initially the computers used to print all in Capital letters. This frequently resulted in a non-standard notation which was difficult not only to read but also to interpret in some cases. This situation has improved recently. Now, it is possible to obtain high speed printing along with the typographical quality of manually typeset material.

Publishers are hoping that one day it will be possible for a computer to automatically compose a page for the newspaper starting from a type written page without any human hand touching it until it is finished and ready for the press.

Machine Aids to Translation of Languages:

Machine translation is becoming a multi-million dollar affair - while the Americans are eager to translate the rapidly increasing number of Russian texts into English and the Russians are trying to translate English texts into Russian, we, Indians, are trying to translate old English texts into the various regional languages. Computers are being programmed for doing

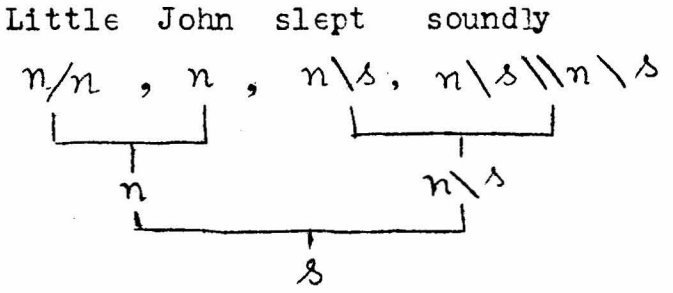
the translation for certain pairs of languages, like, English-Russian, English-French, English-German etc. In the early stages of research in Machine Translation, word-for-word translation devices which made use primarily of a dictionary look-up were proposed. It was found that such attempts lead to an out-put which could be transformed with the help of an expert post-editor into passable translations of the source text. To reduce the burden of the post-editor and to improve the word-for-word translation which was unsatisfactory for certain pairs of languages, it was found that it would be advantageous to analyse the syntactic structure of the source sentences. Y.Bar-Hillel was the first to propose, in 1951, a quasi-arithmatical notation for syntactic description, for a mechanical determination of the constituent structure of a given sentence. He considers the two fundamental catagories for English words to be nominals and sentences, denoted by n and s respectively. His methods is best illustrated by his example of the simple sentence:

Little John slept soundly.

In this sentence "Little" is an adjectival and it will be denoted by n/n (n super n), "slept" is an intransitive verbal which will be denoted by $n\backslash s$ (n sub s), and "soundly" is an intransitive verbal adverbal which will be denoted by $(n\backslash s)\backslash(n\backslash s)$ or $n\backslash s\backslash\backslash n\backslash s$. The mechanical determination of the constituent structure of the sentence is based on the following two self-explanatory rules of cancellation:

$$\alpha,\ \alpha\backslash\beta \rightarrow \beta \quad \text{and} \quad \alpha/\beta\ ,\ \beta \rightarrow \alpha.$$

The symbols representing the word sequence in the sentence will be written down, each symbol being separated from the next one by a comma. A series of symbol sequences, where each sequence results from the preceding sequence by a cancellation rule, is called a derivation. The end result of the derivation is called its exponent. The sentence is said to be well formed if the exponent is a single symbol. Let us write down one of the possible derivations for the simple sentence mentioned above:



From the derivation we note that the exponent is a single, simple symbol and hence the sentence is well formed. Soon, it was realized that this elegant mechanical method of determining the syntactic structure fails in the case of complex sentences, for the following reasons:

For categorizing English words, it is necessary to distinguish various kinds of nominals - viz. singular and plural, animate and inanimate. One should also distinguish between 'declarative' semtences and 'question' sentences. The situation is further complicated since a given word can belong to several categories. For example, the word "that" is often a nominal and used more often as an adjectival. One should not

only take into account all possible categorizations of each word but also consider all the possible derivations before arriving at the exponent to decide whether the sentence is well formed or not. Still worse, it is not clear whether assigning each word to a definite number of categories only would do at all. If it is necessary to assign some words to an infinite number of categories then the mechanical determination of the syntactic structure of a sentence itself breaks down, since there is nc assurance that the number of derivations is finite at all.

To bring to light another drawback of machine translation let us consider the sentence:

The toy box was found in the pen

Assume, for the sake of simplicity that the word "pen" in English has only two meanings, viz. (i) a certain writing appliance and (ii) a small enclosure for various purposes. Every reader with a sufficient knowledge of English will automatically conclude that the word "pen" has the second of the two meanings in the present context. The reason why an intelligent human being is able to pitch upon the correct meaning of the word is because of his knowledge of various other factors (in this context, his knowledge of the relative sizes of the boxes and pens comes to his rescue). But there seems to be no existing or imaginable program which will enable the computer to choose this correct meaning for the word. Even though such

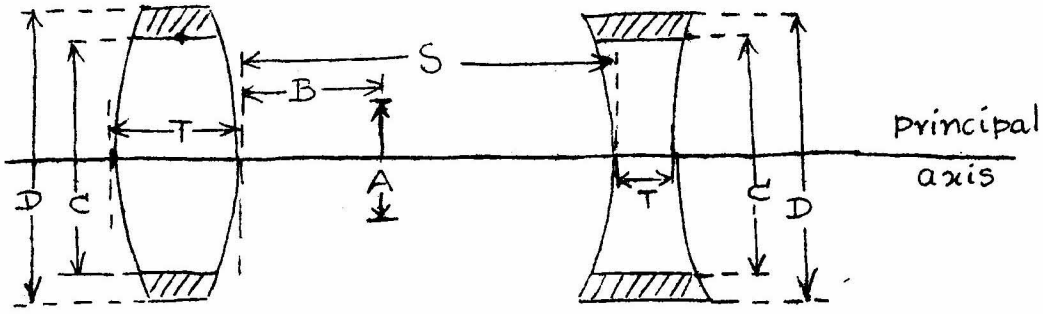
ambiguities may not occur frequently in various documents, the very fact that the possibility of such an occurrence cannot be ruled out points out that Fully Automatic High Quality Translation (FAHQT) is out of question.

Further, the best grammars in any language exhibit lacunae and fail to show the coherence required by the sophisticated computers. The inadequacies of the available grammars to supply the required information has made some research workers ponder as to whether the theory of communication among human beings by means of natural languages itself needs reconsideration and development before satisfactory mechanical translation can be achieved.

Since Fully Automatic High Quality Translation is not yet feasible due to the above mentioned reasons, what is being attempted is to make the reasonably high quality post-edited mechanical word-for-word translations economically (cost-wise) worthwhile. Thus machines are being used as aids to translation.

Optical Design:

We mean by an optical system, a system which consists of mirrors, lenses, and apertures as elements. To illustrate the parameters involved in the design of an optical system let us consider a system of two lenses and a diaphragm or stop.



The design parameters here are:

A - the stop or diaphragm aperture,

B - position of the stop,

C - clear aperture of the lens element,

D - outside diameter of a lens element,

S - separation distance between lens elements,

T - thickness of a lens element,

and besides these the refractive index of glass of which the lens elements are made and the curvature of the lens surfaces are two more design parameters.

The lens designer is given a set of specifications and a list of properties required of the finished product. With the specifications on hand, the traditional lens designer draws on his experience and intuition to select the necessary elements for the design. Then using thin lens equations - viz. equations derived assuming the thickness of the lens to be negligible - he arrives at a rough estimate of the design parameters. The next step is to take the thickness of the lens into account.

Since several procedures recommended by the theoreticians for the transition to the thick lens case are available, he chooses one of them. With the tentative numerical values of the design parameters on hand, the designer comes to grips with the real lens system. Leaving aside the simple-minded concepts used so far, he resorts to complicated and sophisticated methods. He tries to eliminate the Siedel aberrations, viz. Spherical aberration, Coma, Astigmatism, curvature and Distortion, by varying the parameters. This is done in an iterative way. The designer estimates from the size of the aberrations which and by how much the parameters should be changed. Having changed the parameters he recalculates the Siedel aberrations which then provide a starting point for the next set of changes in the parameters. The attempts to reduce the aberrations to sufficiently small values, may take several weeks to several months depending upon the complexity, the skill and the luck of the designer. Now the designer tries to use exact ray tracing formulae to trace the path of light rays from the object field to the image field. The results are plotted on graph paper to provide a visual display of the total aberrations. Once again the designer makes careful adjustments of the design parameters to reduce these aberrations. In this final analysis the lens designer relies on his experience and his bag of tricks which he has acquired during his professional career. These final adjustments can go on for months or even years before the aberrations are reduced to the desired minimum.

The development of the high-speed digital computers proceeded hand-in-hand with its application to optical design. From 1950 onwards, computers have been used for (i) calculating the aberration coefficients and (ii) for automatic ray tracing. Optical design can be thought of as a feed-back process. The optical system at every stage is analysed by ray tracing, providing the designer with the information he needs to improve it. To interpret ray tracing in a realistic manner a method called Spot diagram method has been developed. The rays from a single object point, distributed uniformly over the aperture of the lens, after passing through the lens are intercepted by a plane. The image points on the image plane result in a set of spot diagrams, which can be displayed on a Cathode ray tube. By 1962, almost all optical design in U.S.A. was done on small computers using traditional methods. A few designers used automatic correction for Siedel aberrations. Machines converted good designers into better ones. With the help of the machine the time involved in computing a design was reduced and the computer opened up new vistas for the designer.

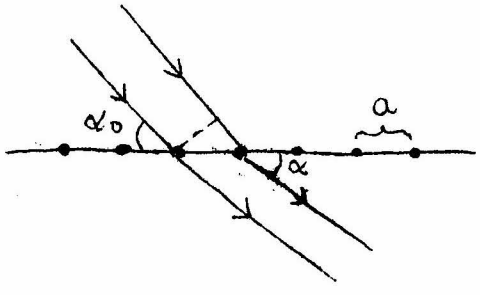
Most ambitious of all proposed routines is that for a fully automatic optical design, which is conceived as a program which will lead to a complete design without the intervention of the operator. Ideally the input would consist only of the design specifications and desired characteristics, leaving the task of deciding the number of lens elements, their refractive

indices and their overall configuration to the computer. At least till 1962, no such automatic optical design program was known to exist.

X-ray Crystallography:

By X-ray crystallography is meant the study of Crystal structure by analysis of X-ray diffraction patterns. A large portion of the research work in this field, over the last 15 or 20 years would not have been possible without the availability of computers. By 1956, routine procedures for the determination and refinement of Crystal structures had been developed and programmed. But since every crystal structure poses its own problem, solving crystal structure continues to be somewhat of an art.

The regular geometrical form of a Crystal is a consequence of the regular arrangement of atoms of which it is built up. When a monochromatic X-ray beam, of wavelength , is incident on a row of equally spaced atoms, the electrons surrounding each atom will serve as sources for scattered waves. These scattered waves will reinforce each other when the path difference for rays scattered by two adjacent atoms in the row corresponds to an integral number of wave lengths.



If α_o is the angle the incident beam makes with the row of atoms (along the X-axis), α the angle the diffracted beam makes with the row of atoms and $\underline{a}$ the atom spacing along the row, (see figure above) then the condition for reinforcement will be:

$$\underline{a}\,(\cos\alpha - \cos\alpha_o) = h\lambda$$

Similarly,

$$\underline{b}\,(\cos\beta - \cos\beta_o) = k\lambda$$

$$\underline{c}\,(\cos\gamma - \cos\gamma_o) = \ell\lambda$$

since the crystal is three-dimensional. These are Laue equations. (h,k,ℓ) are called indices of the diffracted beam or alternatively the indices of a reflection, since they define a plane which reflects the incident beam into the diffracted beam. The experimental data which the crystallographer requires are the intensities of the diffracted beams for most of the (h,k,ℓ) reflections giving significant scattering. These data may vary from 2 or 3 hundred to 20 or 30 thousand intensities for single crystals of varying degrees of complexity. Symmetry elements in the unit cell often cause the intensities of certain orders of (h,k,ℓ) reflections to be absent, and the space group is deduced from these systematic absences, from the equivalence of intensities for reflections and from the variation of the average intensity of the beams with increasing order of diffraction. Locating the positions of the atoms in the unit cell is much more difficult. Indeed, from a computational point of view, the methods of determining approximate

atomic positions in the cell and the refinement of these positions and the atomic shapes comprise most of Crystallography.

It is customary to represent the diffracted ray for each (h,k,ℓ) reflection by a complex number, called the structure factor:

$$F = A+iB = |F|e^{i\alpha}$$

The intensities of the diffracted beam are then proportional to the structure amplitudes $|F|^2$:

$$I(hk\ell) \propto F(hk\ell)\ F^{*}\ (hk\ell) = |F|^2_{hk\ell}$$

A Fourier synthesis first used by Patterson, in 1935, which has had a profound impact in Crystallography, is by means of the function:

$$P(uvw) = \frac{1}{V} \sum_{h} \sum_{k} \sum_{\ell=-\infty}^{\infty} |F|^2_{hk\ell} \exp 2\pi i(hu+kv+\ell w)$$

where V is the volume of the unit cell. A peak in the Patterson function implies the presence of two atoms in the structure separated by the distance of the peak from the Patterson origin and oriented in the same direction. The result of the synthesis is a map which corresponds to the weighted vector set of a set of point atoms. The Patterson synthesis is known to contain all the information needed to solve a Crystal structure but it is often well hidden. A classic example of this Fourier synthesis is in the solution of the structure of Vitamin B_{12}, the most difficult crystal structure

solved so far, by Dorothy Hodgkin and co-workers in 1957.

The International Union of Crystallography compiled in 1962, a world list of all computer programs. These programs are obtained by crystallographers from the authors. Nearly all the crystallographic problems can be conveniently done on a computer which has a 32K memory. A careful job of determining and refining the structure of a monoclinic crystal with 7 to 10 atoms of intermediate difficulty is expected to consume about 5 to 6 hours on a fast machine like CDC 3600 (which we have at the Tata Institute of Fundamental Research, Bombay).

The Plenum Press Series

on

"Symposia on Theoretical Physics and Mathematics"

Edited by ALLADI RAMAKRISHNAN

The proceedings of the Summer Schools and Anniversary Symposia of the Institute are published in a series entitled 'Symposia on Theoretical Physics and Mathematics' by the Plenum Press, New York, a division of the Consultants Bureau Enterprises Inc., Volumes I to X have been published so far.