PROCEEDINGS of the WESTERN JOINT COMPUTER CONFERENCE

held march 1-3, 1955
in los angeles, california
and sponsored jointly by

- The Institute of Radio Engineers
- The American Institute of Electrical Engineers
- The Association for Computing Machinery

Foreword

THE WESTERN Joint Computer Conference has greatly increased in size since the first meeting held in February of 1953. Membership actually has doubled. In 1955 there were 1,490 registrations as compared with 732 in 1953.

The technical programs have always been designed to present material on both analog and digital machines. When possible, field trips to manufacturers and installations have been included in the Conference, as was done in 1955. This year 36 manufacturers exhibited products, as compared with 22 in 1953.

These statistics indicate increasing interest on the part of manufacturers, users, and potential users of the type of automatic data processing equipment discussed. The Conference has changed in character from a meeting of small groups of specialists discussing problems of mutual interest to large meetings involving people from many phases of engineering, management, business control.

These *Proceedings* now are becoming of interest to an increasingly wider spectrum of society.

WILLIAM L. MARTIN
Conference Manager

Proceedings of the 1955 Western Joint Computer Conference

Published by the

INSTITUTE OF RADIO ENGINEERS

1 East 79th Street, New York 21, N. Y.

CONTENTS

1955 Western Joint Computer Conference

OFFICERS

W. L. Martin, Chairman and Conference Manager
Telecomputing Corp.

M. J. Mendelson, Asst. Manager
National Cash Register Co.

W. F. Gunning, Secretary
International Telemeter Corp.

PROFESSIONAL SOCIETY REPRESENTATIVES

IRE	AIEE	ACM
O. Whitby Stanford Research Inst.	G. D. McCann California Inst. of Tech.	D. H. Lehmer Univ. of California, Berkeley
W. L. Martin Telecomputing Corp.	R. R. Bennett Ramo-Wooldridge Corp.	J. L. Barnes Univ. of California, Los Angeles

LOS ANGELES REPRESENTATIVE

J. Salzer
Magnavox Research Lab.

COMMITTEES

Exhibits	Arrangements	Finance
L. L. Kilpatrick, Ch. No. American Aviation, Inc.	G. E. Gourrich, Ch. Telecomputer Corp. J. Tupac, Asso. Ch. The Rand Corp.	R. R. Bennett, Ch. Ramo-Wooldridge Corp.
B. A. Chiappinelli ElectroData Corp.	E. Petry National Cash Register Co.	D. W. Burbeck, Asso. Ch. Ramo-Wooldridge Corp.
P. Kirscher Univ. of California, Los Angeles	J. Siedman National Cash Register Co.	R. J. Barrett Ramo-Wooldridge Corp.
M. Shiowitz National Cash Register Co.	R. Singman Remington-Rand Inc.	L. E. McGauley Ramo-Wooldridge Corp.

Program

G. D. McCann, Chairman
California Inst. of Tech.

R. R. Bennett, H. D. Huskey, O. W. Whitby, E. S. Calhoun, D. N. MacDonald
W. B. Hebenstreit, W. H. Ware, P. Armer, D. T. Greenwood, H. H. Sarkissian

Publicity

E. Tomash, Chairman
Remington-Rand Inc.

R. Bemer
Lockheed Aircraft Corp.

L. W. Cali
ElectroData Corp.

W. Waddell
Librascope, Inc.

E. Weiss
Hughes Aircraft Co.

Publications

A. J. Dowling, Chairman
ElectroData Corp.

Registration

J. E. Stone, Chairman
Components Sales Corp.

B. L. Ettelson
American Machine & Foundry Co.

Trips

C. D. Behm, Chairman
ElectroData Corp.

F. M. Hoar
ElectroData Corp.

V. R. Mackie
Consolidated Engineering Corp.

Exhibitors

Atlas Precision Products Co.
Philadelphia, Pennsylvania

Benson-Lehner Corp.
Los Angeles, California

Beckman Instruments, Berkeley Div.
Richmond, California

Bowmar Instrument Corp.
Fort Wayne, Indiana

Brush Electronics Co.
Cleveland, Ohio

Coleman Engineering Co., Inc.
Los Angeles, California

Computer Control Co., Inc.
Wellesley, Massachusetts

Computer Engineering Associates, Inc.
Pasadena, California

Consolidated Engineering Corp.
Pasadena, California

Eeco Production Co.
Los Angeles, California

ElectroData Corp.
Pasadena, California

Electronic Associates, Inc.
Long Branch, New Jersey

Ferranti Electric, Inc.
New York, New York

Friden Calculating Machine Co., Inc.
Los Angeles, California

G. M. Giannini & Co., Inc.
Pasadena, California

Goodyear Aircraft
Akron, Ohio

Hughes Aircraft Co.
Culver City, California

International Business Machines Corp.
New York, New York

International Telemeter Corp.
Los Angeles, California

Librascope, Inc.
Glendale, California

Magnetic Research Corp.
El Segundo, California

Martin Mann Associates
Los Angeles, California

G. E. Maxon Sales
Culver City, California

F. L. Moseley Co.
Pasadena, California

The National Cash Register Co.
Hawthorne, California

North American Aviation, Inc.
Los Angeles, California

Potter Instrument Co.
Great Neck, New York

J. B. Rea Co., Inc.
Santa Monica, California

Remington-Rand Inc.
New York, New York

C. B. Rush and Associates
Los Angeles, California

Soroban Engineering, Inc.
Melbourne, Florida

Sprague Electric Co.
North Adams, Massachusetts

Teletype Corp.
Chicago, Illinois

Underwood Corp.
Electronic Computer Div.
Long Island City, New York

Wiancko Engineering Co.
Pasadena, California

TUTORIAL SESSIONS PANELS

Subject

Digital Devices and Applications to Science and Engineering

Panel Members

P. ARMER, Chairman
The Rand Corporation

W. F. BAUER
Ramo-Wooldridge Corp.

J. P. NASH
University of Illinois

K. POWELL
Babcock and Wilcox

E. C. YOWELL
National Cash Register Co.

Subject

Analog Computers

Panel Members

D. T. GREENWOOD, Chairman
Lockheed Aircraft Corp.

W. J. DIXON
Computer-Engineering Asso.

C. H. WILTS
California Institute of Technology

N. L. IRVINE
Aerojet-General Corp.

L. G. WALTERS
University of California, Los Angeles

DISCUSSION SESSIONS PANELS

Subject

Digital Systems and Applications to Business

Panel Members

E. S. CALHOUN, Chairman
Stanford Research Institute

O. H. NIELSON
Stanford Business School

R. D. DOTTS
Pacific Mutual Life Insurance Co.

R. H. WAGNER
ElectroData Corp.

Subject

Techniques of Handling Non-Uniform Length Data

Panel Members

H. H. SARKISSIAN, Chairman
National Cash Register Co.

G. E. GOURRICH
Telecomputing Corp.

J. A. BRUSTMAN
Radio Corp. of America

R. W. MURPHEY
International Business Machines

H. F. MITCHELL
Remington Rand

Subject

Language and Communication, Common Language

Panel Members

D. N. MACDONALD, Chairman
ElectroData Corp.

D. W. PENDERY
International Business Machines

E. M. MCCORMICK
Naval Ordnance Lab., Corona

W. A. KAUTZ
Stanford Research Institute

L. W. CALKINS
U. S. Steel

Transfer-Function Synthesis with Computer Amplifiers and Passive Networks

M. V. MATHEWS† AND W. W. SEIFERT†

Summary—**The study of dynamic systems on an analog computer often involves the synthesis of complex transfer functions. Techniques from the field of network synthesis are combined with methods used in electronic-differential-analyzer work to provide effective means for realizing these transfer functions with a minimum of computer equipment. The basic ideas of associating high-gain amplifiers and phase-inverting amplifiers with resistor-capacitor networks are applied in order to obtain three systematic methods for synthesizing transfer functions of various degrees of complexity. The methods described are illustrated by an example.**

Introduction

MANY analog-computer problems require the synthesis of complex transfer functions. Realization of these transfer functions with a minimum of computing equipment is a practical objective that can be achieved by the use of passive networks in combination with computer amplifiers.

Procedures for realizing stable transfer functions with passive networks containing resistances, inductances, and capacitances have been described by Darlington,[1] Guillemin,[2] and others, but in many instances these methods have practical limitations. A synthesis involving inductors frequently is complicated by inherent losses and distributed capacitances. Furthermore, inductors, particularly when used at the low frequencies encountered in computers and control systems, tend to be large and expensive.

Highly versatile methods, which avoid the limitations inherent in passive-network synthesis, may be derived by associating active elements such as amplifiers with passive networks. The basic ideas of associating high-gain amplifiers and phase-inverting amplifiers with passive networks were developed from feedback-amplifier theory[3] and were discussed by Blackman, Bode, and Shannon in a classified report in 1946 and by Belove[4] in 1950. In the computer field, full advantage has not been taken of the potentialities of these methods because they have not been developed into a systematic synthesis procedure that uses computer amplifiers as the active elements.

This paper describes three systematic synthesis methods which are particularly adaptable to computer applications. First, the well-known method that employs one amplifier and a feedback network is presented in its general form. Next, an original 3-amplifier design that offers distinct advantages in the synthesis of complicated functions is described. The third method presented is a generalized version of the basic technique used for the solution of differential equations on a differential analyzer. In this last scheme, the active elements are integrators rather than amplifiers.

The computer amplifiers utilized in the synthesis procedures are assumed to be ideal amplifiers which draw zero input current, have zero output impedance, and provide any desired real gain. The assumption of a perfect amplifier is valid in the frequency range from direct current through the audio spectrum because, for these frequencies, computer amplifiers have been perfected to such a degree that their deviations from the ideal are as small as the parasitic errors in resistors and capacitors.

Synthesis Using One Feedback Amplifier

In electronic differential analyzers, integration is performed by associating a simple rc network with a high-gain amplifier, as shown in Fig. 1. If an ideal amplifier with infinite gain is assumed, then analysis of the circuit yields for the ouput voltage

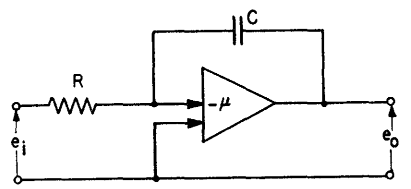


Fig. 1—Block diagram for integrator used in electronic differential analyzers.

$$e_0 = -\left(\frac{1}{RC}\int_0^t e_i dt + E_0\right), \tag{1}$$

where E_0 is the value of the voltage across the capacitor at $t=0$. These same simplifications give for the transfer function of this circuit

$$\frac{e_0}{e_i} = -\frac{1}{RCs}, \tag{2}$$

where s is the complex-frequency variable.

Fig. 2 shows the simplest generalization of the basic integrator arrangement of Fig. 1. If the same simplifying assumptions used to derive (1) and (2) are applied to

† Massachusetts Institute of Technology, Cambridge, Mass.

[1] S. Darlington, "Synthesis of reactance 4-poles which produce prescribed insertion loss characteristics," *Jour. Math. Phys.*, vol. 18, pp. 257–353; September, 1939.

[2] E. A. Guillemin, "A Summary of Modern Methods of Network Synthesis," Advances in Electronics, vol. III, Academic Press, Inc., New York, pp. 261–303; 1951.

[3] H.W. Bode, "Network Analysis and Feedback Amplifier Design," D. Van Nostrand Co., Inc., New York, N. Y., pp. 226–248; 1945.

[4] C. Belove, "Synthesis of Active Low-Frequency Networks," Navord Report No. 1491, Bureau of Ordnance, Washington, D. C.; November, 1950.

the analysis of Fig. 2, the transfer function of the circuit is found to be

$$\frac{e_0}{e_i} = -\frac{Y_A}{Y_B}. \qquad (3)$$

If Y_A and Y_B are 2-terminal rc networks, all their poles and zeros must alternate along the negative real axis of the complex-frequency plane and the lowest critical frequencies must be zeros. Consequently, the poles and zeros of the transfer function also must lie on this axis, but two poles or two zeros may occur together, and the lowest critical frequency may be a pole. Any transfer functions meeting these conditions can be written in the form

$$\frac{e_0}{e_i} = -\frac{\dfrac{N(s)}{G(s)}}{\dfrac{D(s)}{G(s)}}, \qquad (4)$$

where $G(s)$ can be selected so that $N(s)/G(s)$ and $D(s)/G(s)$ can be realized as 2-terminal rc networks.

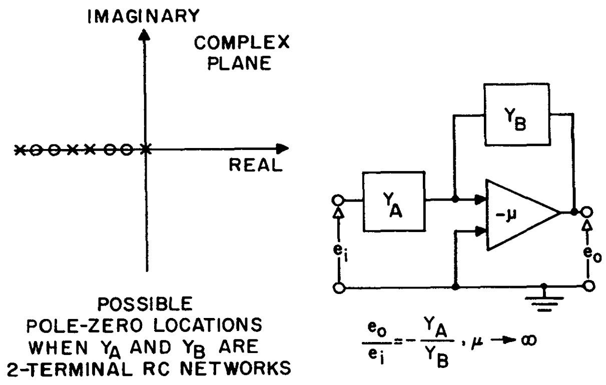

IMAGINARY
COMPLEX PLANE
REAL
POSSIBLE POLE-ZERO LOCATIONS WHEN Y_A AND Y_B ARE 2-TERMINAL RC NETWORKS
Y_B
Y_A
$-\mu$
e_i
e_o
$\frac{e_o}{e_i} = -\frac{Y_A}{Y_B}, \mu \to \infty$

Fig. 2—Block diagram for 1-amplifier realization with 2-terminal networks.

The synthesis of Y_A can be carried out in several ways, one of the simplest being to expand $N(s)/G(s)$ in the form

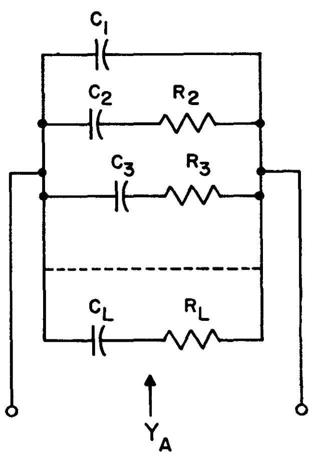


Fig. 3—Form of network.

$$\frac{N(s)}{G(s)} = s\left[C_1 + \sum_{i=2}^{Q} \frac{\dfrac{1}{R_i}}{s + \dfrac{1}{R_iC_i}}\right], \qquad (5)$$

where

$$C_1 = \lim_{s\to\infty}\left[\frac{1}{s}\,\frac{N(s)}{G(s)}\right]. \qquad (6)$$

The sum

$$\sum_{i=2}^{Q} \frac{\dfrac{1}{R_i}}{s + \dfrac{1}{R_iC_i}} \qquad (7)$$

is obtained by making a partial-fraction expansion of

$$\frac{1}{s}\left[\frac{N(s)}{G(s)} - C_1 s\right]. \qquad (8)$$

The resulting network is shown in Fig. 3, where the values of R_i and C_i are in ohms and farads.

Substitution of 3-terminal networks in place of the 2-terminal networks of Fig. 2 yields a useful generalization of this method of synthesis. The resulting circuit, illustrated in Fig. 4, can be analyzed in terms of the

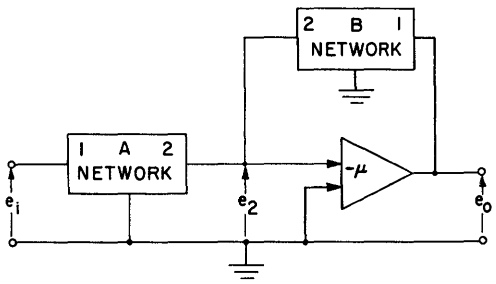


Fig. 4—Block diagram for 1-amplifier realization with 3-terminal networks.

input, output, and transfer admittances of the network. These admittances are defined for the A network by the relationships

$$i_1 = Y_{A11}e_1 - Y_{A12}e_2, \qquad (9)$$

and

$$i_2 = -Y_{A12}e_1 + Y_{A22}e_2 \qquad (10)$$

where the currents and voltages are shown in Fig. 5. A similar definition applies to the B network. The volt-

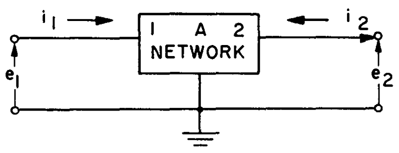


Fig. 5—Definition of admittances.

ages in the circuit of Fig. 4 are related by the equation

$$e_2(Y_{A22} + Y_{B22}) = e_i(Y_{A12}) + e_0(Y_{B12}). \tag{11}$$

Furthermore,

$$e_0 = -\mu e_2. \tag{12}$$

Solution of (11) and (12) yields for the transfer function e_0/e_i

$$\frac{e_0}{e_i} = -\frac{Y_{A12}}{Y_{B12} + \dfrac{Y_{A22} + Y_{B22}}{\mu}}. \tag{13}$$

As μ becomes infinite, e_0/e_i approaches the negative of the ratio of the transfer admittances,

$$\frac{e_0}{e_i} = -\frac{Y_{A12}}{Y_{B12}}. \tag{14}$$

The error caused by a finite μ can be evaluated from (13), which is the exact expression for the realized transfer function. The errors can be determined either as the displacements of the poles of the realized transfer function from the desired poles or as the error in the amplitude and phase of the realized transfer function at real frequencies. To keep the error small, it is necessary that at all frequencies

$$Y_{B12} \gg \frac{Y_{A22} + Y_{B22}}{\mu}. \tag{15}$$

At high frequencies, either or both of the output admittances Y_{A22} and Y_{B22} may tend to become infinite. If such is the case, the B network should be so designed that Y_{B12} also goes to infinity at high frequencies.

The transfer admittance of a 3-terminal network formed entirely of resistances and capacitances can have only simple poles[5] which must lie on the negative real axis of the complex-frequency plane but may have zeros which lie anywhere in the complex-frequency plane except on the positive real axis and which need not be simple. The poles of e_0/e_i in (14) follow from the poles in Y_{A12} or from the zeros of Y_{B12}, while the zeros of e_0/e_i follow from the zeros of Y_{A12} or from the poles of Y_{B12}. Consequently, little theoretical restriction is placed on the type of transfer impedance that can be formed by using a circuit of the type shown in Fig. 4.

Several general procedures for synthesizing 3-terminal rc networks have been given in the literature.[5,6] These procedures are too lengthy to include here, but an example employing one of the methods[6] is given later.

The principal restrictions imposed on this realization are the complexity of the synthesis calculations, the large number of elements, and large range of element values which may be required.

[5] A. Fialkow and I. Gerst, "The transfer function of general two terminal-pair rc networks," *Quart. Appl. Math.*, vol. 10, pp. 113–127; April, 1952.

[6] E. A. Guillemin, "Synthesis of rc-networks," *Jour. Math. Phys.*, vol. 28, pp. 22–42; April, 1949.

Realization with Three Amplifiers

Although few theoretical limitations are imposed on the type of transfer function realizable with the single-feedback-amplifier method just described, the use of additional amplifiers permits increased flexibility in the realization of the transfer function. This flexibility can reduce the number of passive elements required to obtain a given function, decrease the spread of element values, and simplify the synthesis calculations. Such expedients are particularly important when complicated functions with many poles must be realized.

One synthesis method using three amplifiers is developed to demonstrate that any transfer function can be realized in this way. Once the particular method is understood, many possible variations become obvious.

The circuit for the 3-amplifier realization is shown in Fig. 6. This circuit differs from the 1-amplifier realiza-

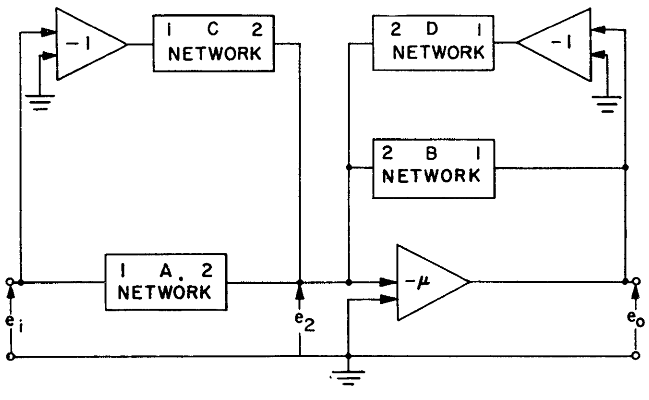


Fig. 6—Block diagram for 3-amplifier realization.

tion shown in Fig. 2 only by the addition of the C and D networks and the inverting amplifiers driving these networks. As will be brought out in the discussion, 2-terminal networks are sufficient to realize any transfer function; hence, this case is considered.

The voltages in the system obey the relations

$$e_2(Y_A+Y_B+Y_C+Y_D) = e_i(Y_A - Y_C) + e_0(Y_B - Y_D), \tag{16}$$

$$e_0 = -\mu e_2. \tag{17}$$

Solution of (16) and (17) yields for the transfer function from e_i to e_0

$$\frac{e_0}{e_i} = -\frac{Y_A - Y_C}{Y_B - Y_D + \left(\dfrac{Y_A + Y_B + Y_C + Y_D}{\mu}\right)}. \tag{18}$$

As μ becomes large, (18) assumes the limiting form

$$\frac{e_0}{e_i} = -\frac{Y_A - Y_C}{Y_B - Y_D}. \tag{19}$$

If the desired transfer function is expressed as a ratio of two polynomials $N(s)/D(s)$, the admittances must satisfy the relation

$$\frac{Y_A - Y_C}{Y_B - Y_D} = \frac{N(s)}{D(s)}. \tag{20}$$

In order to realize the admittances as rc networks, (20) is separated to give

$$Y_A - Y_C = \frac{N(s)}{G(s)} \tag{21}$$

and

$$Y_B - Y_D = \frac{D(s)}{G(s)}, \tag{22}$$

where $G(s)$ is an arbitrary polynomial which does not alter the realized transfer function.

The realization follows the method used to obtain Y_A and Y_B in Fig. 2. The fraction $N(s)/G(s)$ is expanded in the series given by (5), and the terms in the resulting expansion are divided between the A and C networks so all the elements have positive values. The additional freedom gained from allowing negative terms in the expansion makes possible the realization of any $N(s)/G(s)$ with 2-terminal rc networks, provided that the two following conditions are met: 1) The zeros of $G(s)$ lie on the negative real axis; 2) The ratio $N(s)/G(s)$ goes to infinity no faster than s as s becomes infinite. An identical procedure is used to realized $D(s)/G(s)$.

The error introduced by a finite gain μ can be evaluated from (18), which is the exact expression for the realized transfer function. The method is the same as the method already described for evaluating the errors in the 1-amplifier realization. However, in (18) the possibility exists of changing the synthesis procedure slightly to realize exactly the desired transfer function with a finite μ. Relation (18) may be rewritten in the form

$$\frac{e_0}{e_i} = \frac{Y_A - Y_C}{Y_B\left(1+\frac{1}{\mu}\right) - Y_D\left(1-\frac{1}{\mu}\right) + \frac{1}{\mu}(Y_A+Y_C)}. \tag{23}$$

If the desired transfer function is again designated $N(s)/D(s)$, it can be realized by making

$$Y_A - Y_C = \frac{N(s)}{G(s)} \tag{24}$$

and

$$Y_B\left(1+\frac{1}{\mu}\right) - Y_D\left(1-\frac{1}{\mu}\right) = \frac{D(s)}{G(s)} - \frac{1}{\mu}(Y_A+Y_C). \tag{25}$$

If $\mu > 1$, the A, B, C, and D networks can always be realized as 2-terminal rc networks by using expansions of the form given in (5). The exact realization, obtained at the expense of including additional elements in the B and D networks, is justified only in special instances because the errors caused by a finite μ in the approximate realization are usually less than the errors due to parasitic behavior of the elements.

A major advantage of the 3-amplifier synthesis procedure is the simplicity of the calculations required to obtain the element values. The spread of element values is determined by the spread of the terms in the expansion of $N(s)/G(s)$ given in (5) and the corresponding expansion of $D(s)/G(s)$. The arbitrary zeros of $G(s)$ can be chosen by a trial-and-error approach to control this spread.

Synthesis with Integrators

Another synthesis approach, different from the two already described, consists of combining a number of simple transfer functions to produce a desired function. A transfer function of the form.

$$F(s) = \frac{e_0}{e_i} = \frac{a_m s^m + a_{m-1}s^{m-1} + \cdots + a_0}{s^m + b_{m-1}s^{m-1} + \cdots + b_0}, \tag{26}$$

where the a's and b's are any real constants, can be instrumented as shown in Fig. 7. This method, often

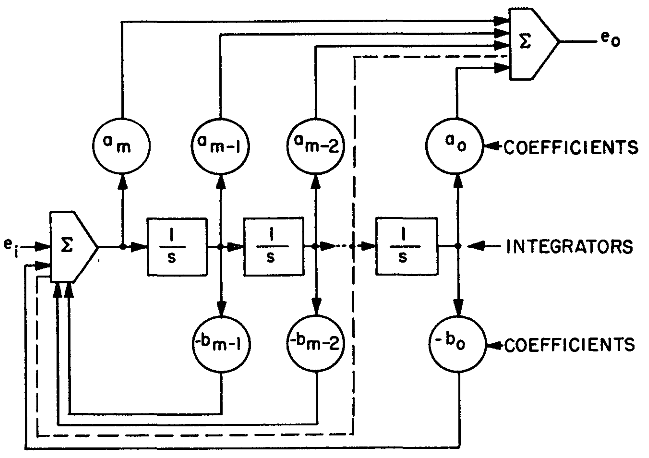


Fig. 7—Integrator realization.

applied in analog-computer work, is particularly suitable for realizing transfer functions having a small number of poles, particularly if the coefficients in the function require frequent change. Since the coefficients appear directly as the gains of amplifiers, almost no calculations are required in the synthesis.

The principal errors generated by this realization stem from the limited frequency range over which the physical integrators approximate true integrators. Although in this method the departure from ideality restricts the frequency range over which a transfer function may be realized, such a limitation presents no serious drawback in many computer applications. A more important consideration is that if high-order functions are to be synthesized, an excessively large number of active units is required, and the equipment reduction effected by using the methods discussed in the previous sections assumes practical significance.

Illustrative Examples

The following equation typifies the complexity of the transfer functions frequently encountered in the analysis of systems where an analog-computer representation may be desired:

$$\frac{e_0}{e_i} = -\frac{(s+1)\left(\frac{1}{8}s+1\right)}{(4s+1)\left(\frac{1}{16}s+1\right)\left(\frac{s^2}{16}+\frac{1}{8}s+1\right)}\cdot \tag{27}$$

To illustrate the 1-amplifier and 3-amplifier synthesis procedures, this function is realized by each of the methods. Synthesis with integrators is so direct that it will not be illustrated. The function was selected not to represent any particular type of system but rather to give an example which is neither trivial nor impracticable to realize by either of the methods.

For 1-amplifier synthesis, the function of (27) is put into the form

$$\frac{e_0}{e_i} = -8\,\frac{\dfrac{1}{\left(s+\frac{1}{4}\right)(s+16)}}{\dfrac{s^2+2s+16}{(s+1)(s+8)}}\cdot \tag{28}$$

The A network then is synthesized by the method of Guillemin[6] to have a transfer admittance

$$Y_{A12} = \frac{1}{\left(s+\frac{1}{4}\right)(s+16)} \tag{29}$$

by realizing a driving-point admittance

$$Y_{A11} = \frac{\left(s+\frac{1}{8}\right)(s+8)}{\left(s+\frac{1}{4}\right)(s+16)} \tag{30}$$

in such a form that both zeros of Y_{A12} occur at infinity. The zeros of Y_{A11} are arbitrary except that they must be selected so that Y_{A11} can be realized as the input admittance of an rc network. The B network is synthesized to have a transfer admittance

$$Y_{B12} = \frac{s^2+2s+16}{(s+1)(s+8)} \tag{31}$$

by connecting in parallel three networks with the short-circuit transfer admittances

$$Y_{B12}{}^{(0)} = \frac{16}{(s+1)(s+8)}, \tag{32}$$

$$Y_{B12}{}^{(1)} = \frac{2s}{(s+1)(s+8)}, \tag{33}$$

$$Y_{B12}{}^{(2)} = \frac{s^2}{(s+1)(s+8)}\cdot \tag{34}$$

The driving-point admittances of the networks are selected as

$$Y_{B11}{}^{(0)} = \frac{\left(s+\frac{1}{2}\right)(s+4)}{(s+1)(s+8)} \tag{35}$$

and

$$Y_{B11}{}^{(1)} = Y_{B11}{}^{(2)} = \frac{s(s+4)}{(s+1)(s+8)}, \tag{36}$$

and the realizations are carried out to control the zeros of the transfer admittances. The resulting realization is shown in Fig. 8, where the impedance of the networks

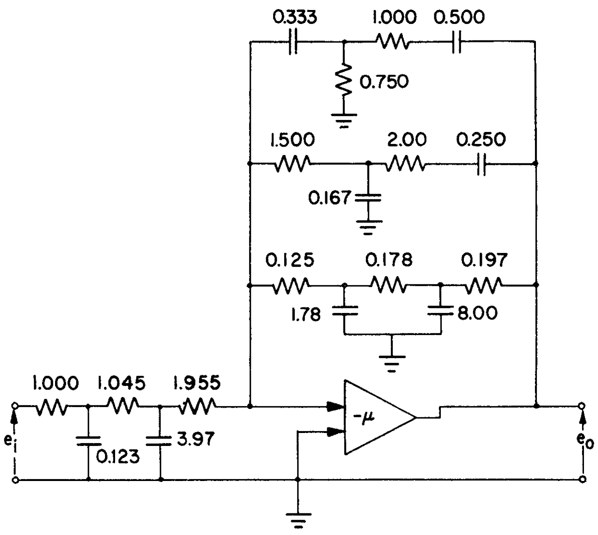


Fig. 8—Example of 1-amplifier realization. Element values in megohms and microfarads.

has been increased by a factor of 10^6, and the realized transfer function differs from the specified transfer function by a constant gain factor of $\frac{1}{8}$. The number of required passive elements is 18; the range in capacitor values is 48 to 1, and the range in resistor values is 16 to 1.

The 3-amplifier synthesis can be carried out effectively by selecting

$$G(s) = (s+1)(s+8)(s+16). \tag{37}$$

Substitution of this particular $G(s)$ along with $N(s)$ and $D(s)$ from (27) into (21) and (22) gives, except for a constant multiplying factor,

$$Y_A - Y_C = \frac{1}{s+16} \tag{38}$$

and

$$Y_B - Y_D = \frac{\left(s+\frac{1}{4}\right)(s^2+2s+16)}{(s+1)(s+8)}\cdot \tag{39}$$

The right-hand sides of (38) and (39) now may be expanded in partial-fraction forms as

$$Y_A - Y_C = 0.0625 - \frac{0.0625s}{s+16} \tag{40}$$

and

$$Y_B - Y_D = s + \frac{1}{2} + \frac{1.61s}{s+1} - \frac{8.86s}{s+8}, \tag{41}$$

which may be realized as shown in Fig. 9. The imped-

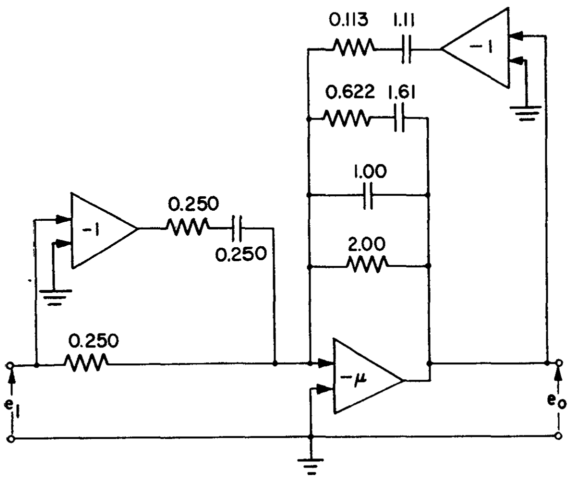


Fig. 9—Example of 3-amplifier realization. Element values in megohms and microfarads.

ances of the A and C networks have been increased by 15.6×10^3, and the impedances of the B and D networks by 10^6. The realized transfer function differs from the specified transfer function by a constant gain

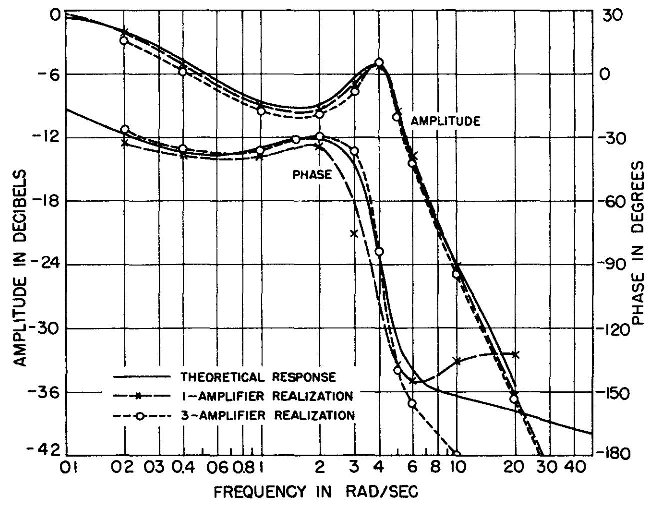


Fig. 10—Response data.

factor of 8. The number of required passive elements is 9. The range in capacitor values is 6.43 to 1, and the range in resistor values is 17.7 to 1. The range of element values for this example is so small that no practical difficulty arises in building the networks; consequently, no necessity exists for reducing the spread by adjusting $G(s)$.

To test the synthesis procedures experimentally, the realizations shown in Figs. 8 and 9 were constructed from paper capacitors and carbon resistors. Element sizes were adjusted to be within 1 per cent of the required values. The measured amplitude and phase characteristics of the realizations together with the characteristics computed from (27) are shown in Fig. 10. Agreement between the measured curves and the desired characteristics demonstrates the soundness of the synthesis. At high frequencies the deviations in the phase curves fall within the limits of error of the experimental procedure employed.

Conclusions

The example shows that either of the two synthesis procedures illustrated in this paper is practical for realizing a 4-pole transfer function. The 1-amplifier method utilizes a minimum of active equipment but requires the more complex synthesis calculations and a greater number of passive elements. If more complex transfer functions are to be realized, the passive networks associated with the 1-amplifier realization become impractical because of the number of elements and the spread of element values. Therefore, this method is most suitable for realizing transfer functions no more complicated than (27). The 3-amplifier realization requires more active equipment but simplifies the synthesis calculations and reduces the complexity of the passive networks. As the complexity of the transfer function increases, the advantages of the method rapidly become more apparent.

If sufficient active equipment is available, the third method, integrator realization, may be used. The calculations for this synthesis are the simplest of the three methods, and the coefficients of the transfer function may be changed easily because they appear directly as amplifier gains. The three methods present a systematic approach to the realization of a wide variety of transfer functions which cannot be obtained practicably by using only passive elements.

Simulation by Modeling

N. L. IRVINE† AND L. DAVIS†

Introduction

SIMULATION by modeling may be classified into three related methods.

1. We may use analog models which obey the same laws as the phenomena we wish to study. Instruments such as network analyzers, slide rules, and electrolytic tanks are examples of devices that are used to make analog models.

2. We may use mathematical models to describe phenomena we wish to study. Quite often we resort to high-speed computing machines or differential analyzers to solve specific problems from the equations derived in our mathematical models.

3. We may subject scaled models of equipment to actual or simulated environments. Wind-tunnel testing of airfoils is a notable example. Actual equipment may also be subjected to simulated expected or known environments, for example, experimentation with personnel and equipment in high-altitude chambers.

Many of our complex phenomena may be studied effectively by simulation. The design of multidimension filters, sometimes called space filters, may be studied in this manner.[1] We are all familiar with the problem of detecting a message from an electrical signal which contains the message and noise. Suppose we wanted to detect a particular geometric shape in a field which contained a large number of configurations as well as the desired form. Let us choose the letter *Z* as our geometric shape, surrounded by a number of other configurations; we could detect our letter *Z* by surveying the entire field with a filter which had the letter *Z* detailed to give full transmission while the response of the remainder of the filter gave small transmission. Obviously, when the configuration of the filter matches the desired information, a high degree of transmission is obtained, compared with that obtained from a nonmatching condition. If we were to reverse the *Z* to $\reflectbox{Z}$ we would not be able to find our message.[2]

A special case of this type of filtering is that in which a small aperture is used to scan the field. This technique may in general be treated as a unidimensional process, and it has been rather thoroughly explored for many scanning devices, such as equipment for the facsimile process and for television.

In general, space filtering may be extended to *n* dimensions, where color, intensity, and location as well as geometric shape may be introduced. The analogy between the unidimensional electronic-filter problem and the multidimensional filter problem is quite complete. The methods of analysis of space filtering are an extension of the usual techniques for dealing with filters in electronic circuits. In both cases, the output from the filter is a linear weighted average of the input information; thus, the weighting function of a filter is determined. In both cases, the problems are as follows:

1. To design a method for searching for a given type of signal.
2. To reproduce a signal, with discrimination in favor of a desired signal and against others.
3. To improve a signal, i.e., to remove distortion.

Discussion

In electronic circuitry, the input and output messages are described in terms of frequency response and frequency content; thus, the independent parameter is time. With certain restrictions, any single-valued function of time has a Fourier transform as a function of frequency. Similarly, any general field distribution in space has a Fourier transform, but since the space distribution is a function of lengths, the transformed distribution is a function of reciprocal lengths or wave numbers. Rapid changes of the function in space will give relatively large high-wave-number components in the spectrum, just as rapid changes of amplitudes of electrical signals give large high-frequency components.

A unidimensional field distribution will have a unidimensional Fourier transform. In the case of a planar field distribution, the transformed function will be of two wave-number variables where the two-component position vector forms a two-component wave-number vector.

Let us look more closely at the similarity between the transforms used in electronic filter design and the *n*-dimensional transforms. The transform of $f(x)$ is as follows:

$$F(k) = \int_{-\infty}^{\infty} f(x) e^{ikx} dx \tag{1}$$

and under certain conditions

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(k) e^{-ikx} dk. \tag{2}$$

For the *n*-dimensional field, we define the field position vector $\bar{R}$, and it can be shown that under suitable conditions the function $f(\bar{R})$ has a Fourier transform

$$F(\bar{k}) = \int_{-\infty}^{\infty} f(\bar{R}) e^{i\bar{K}\cdot\bar{R}} d\bar{R}, \tag{3}$$

† Aerojet General Corp., Azusa, Calif.

[1] P. Elias and D. S. Grey, "Fourier treatment of optical processes," *Jour. Opt. Soc. Amer.*, vol. 42, p. 127; February, 1952.

[2] R. C. Jones, "Detection of Targets Against the Sky Background," Rep. 535, Polaroid Corp., Cambridge, Mass.; April 12, 1954.

where $\overline{K}\cdot\overline{R}$ is the usual dot product notation for vector operations, and for an n-dimensional Cartesian coordinate system

$$\overline{K}\cdot\overline{R} = k_1x_1 + k_2x_2 + \cdots \tag{4}$$

And (3) is

$$F(k_1, k_2 \cdots k_n)$$
$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} f(x_1x_2\cdots x_n)e^{i\overline{K}\cdot\overline{R}}d_{x_1}d_{x_2}\cdots d_{x_n}. \tag{5}$$

A rigorous treatment may be found in several texts.[3]

Let us confine our attentions to the two-dimensional case and look at some commonly used functions transformed in one dimension and then in two dimensions (shown in Figs. 1 and 2).

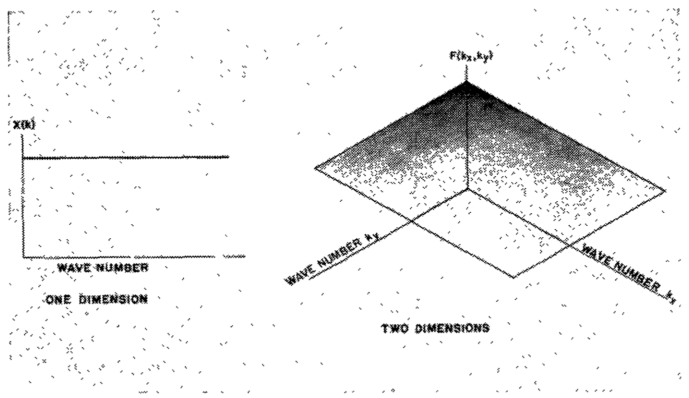


Fig. 1—Transform of the delta or impulse function.

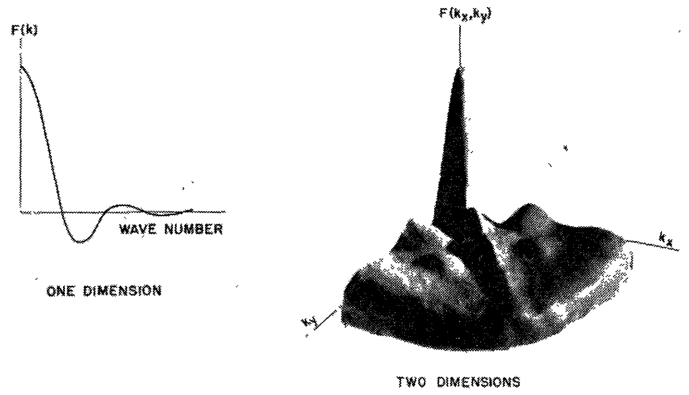


Fig. 2—Transform of a square pulse.

Suppose we have a two-dimensional signal, $J(x, y)$, which contains the desired message, $S(x, y)$, and an unwanted component $N(x, y)$. From this signal we wish to detect, discriminate, and reproduce this message as accurately as possible. Generally, we cannot optimize both the detection and discrimination characteristics at the same time, since an optimum detector leads to a filter design which maximizes signal-to-noise ratio and optimum discrimination implies the minimizing of the mean square error[4] between the desired message and the actual output message from the filter.

We may define the filter weighting function $W(x, y)$ in a manner similar to that in electrical theory. Then, within the restrictions mentioned before, the Fourier transformations are valid and we can define the autocorrelation function, $\phi(x, y)$, and the power spectrum, $I(k_x, k_y)$:

$$\phi(x', y') = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W(x, y)W(x + x', y + y')dxdy \tag{6}$$

and

$$I(k_x, k_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{i(k_xx+k_yy)}\phi(x, y)dxdy. \tag{7}$$

As in electronic analysis, the Fourier treatment is limited to linear operations. A linear operation upon a field distribution function of an n-dimensional argument may be defined as one which replaces the value of the function at a point by a linear weighted average taken over the neighborhood at that point.

If we want to examine a field for a particular type of distribution, we must, for practical reasons, introduce some sort of scanning technique. Scanning usually implies a time variation in one or more of the n-dimensions.

We might let x vary with time and scan a large area bit by bit. An example of this type of scanning is the facsimile process. Scanning may also be done by rotation, as in the early television cameras. On the other hand, if we add detail to the filter where the transmission is a function of x and y, we may sample much larger fields and even the entire field of interest at once.

If we examine a field distribution, $J(x, y)$, with a filter whose weighting function is $W(x, y)$, and we wish to determine mathematically the output from the filter, we must perform the following integrations:

$$E(x', y') = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} J(x, y)W(x + x', y + y')dxdy, \tag{8}$$

where the integration limits may take on values dependent on the field boundaries. We immediately recognize this as a form of the convolution integral and we can transform (8) to wave-number space:

$$E'(k_x, k_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x, y)e^{i(k_xx+k_yy)}dxdy \tag{9}$$
$$= J'(k_x, k_y)W'(k_x, k_y).$$

If we allow x to vary with time

$$x = vt, \tag{10}$$

then

$$vk_x = \omega, \tag{11}$$

[3] I. N. Sneddon, "Fourier Transforms," McGraw-Hill Book Co., Inc., New York, N. Y.; 1951.

[4] N. Wiener, "The Extrapolation, Interpolation, and Smoothing of Stationary Time Series," Technology Press and John Wiley & Sons, Inc., New York, N. Y.; 1949.

and (9) becomes

$$E'\left(\frac{\omega}{v}, k_y\right) = J'\left(\frac{\omega}{v}, k_y\right) W'\left(\frac{\omega}{v}, k_y\right). \quad (12)$$

At this point we must pause and consider the nature of the distribution function

$$J'\left(\frac{\omega}{v}, k_y\right).$$

It will generally be the superposition of two separate contributions denoted for convenience by

$$J_S'\left(\frac{\omega}{v}, k_y\right) \text{ and } J_N'\left(\frac{\omega}{v}, k_y\right),$$

where the S subscript refers to the desired spectrum and N to the undesired background. Then the output spectrum is also resolvable into separate terms

$$E_S'\left(\frac{\omega}{v}, k_y\right) \text{ and } E_N'\left(\frac{\omega}{v}, k_y\right).$$

Optimization of the filter function $W'(\omega/v, k_y)$, then must involve some criterion, such as maximizing the ratio of

$$\frac{|E_S'|}{|E_N'|},$$

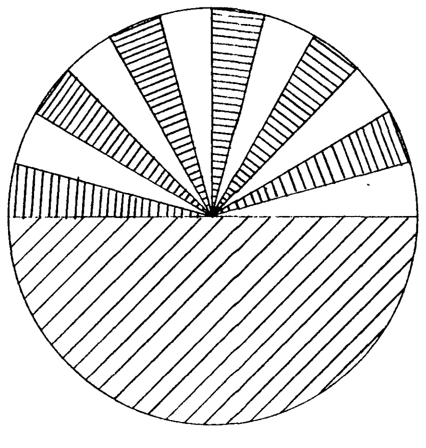
at least over some particular region in $(\omega/v, k_y)$, and the exact form of W' will depend on the criterion used.

Let us now consider a particular system of considerable practical interest. Let us suppose that we wish to detect a round source of radiation surrounded by a general distribution of radiation such as a clouded sky. We now are concerned with the problem of enhancing the signal from the round source, J_S, as compared with general background distribution, J_N. To do this, we may scan the field with an appropriate optical system making use of a filter whose weighting function is W. An example of a weighting function[5] to be used is shown in Fig. 3.

For many possible forms of J_N, which are of great interest, the computation of the transforms in closed form is extremely difficult. The W function which would be determined as optimum for a particular configuration of J_S and J_N would no longer be so for a J_S translated with respect to J_N. The computation involved in attempting to optimize such a space filter with purely analytical tools would be enormous.

A convenient alternative to this head-on analytical attack is to use the analysis to determine the significant features of the J_S and J_N distributions and then replace them by simplified models which can be conveniently simulated by optical-electrical-mechanical means. With such a scheme, a change in some parameter such as position or intensity is a matter of a simple displacement or the addition of an optical attenuator.

[5] R. C. Jones, "On the Theory of Scanning Reticles (Episcotesters)," Rep. 542, Polaroid Corp., Cambridge, Mass.; May 25, 1954.



▨ Partial or Zero Transmission

□ 100% Transmission

Fig. 3—Typical n-wedge optical filter—rotational scan.

Such a simulation would require a system incorporating the following features:

1. Capable of forming a real planar image of the simplified function model.
2. Accessibility of this focal plane to an experimental space filter element with the desired scanning motion.
3. An appropriate type of integrating photometer to measure the transmitted light.
4. For a scanning motion, the information is converted into a function of time; thus adequate electronic processing equipment is required.

A schematic of such a simulator is shown in Fig. 4.

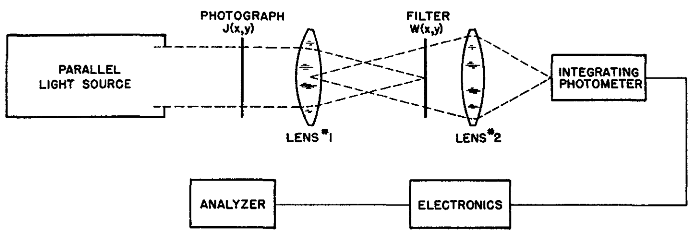


Fig. 4—Schematic of a space filter simulator.

In the operation of this simulator, the parallel light illuminates the field photograph $J(x, y)$. Lens 1 focuses the image of $J(x, y)$ on the filter $W(x, y)$, and Lens 2 serves as a collector which concentrates all the energy transmitted by the filter onto the integrating photometer. From the photometer, we have an electric signal which is analyzed principally for frequency content. The scanning operation may be performed in three ways:

1. Movement of the field photograph, $J(x, y)$.
2. Movement of the filter, $W(x, y)$,
3. A combination of both.

The scanning operation depends upon the particular problem at hand. The motions may, for example, be linear or circular.

In our particular problem, we may prepare photo-

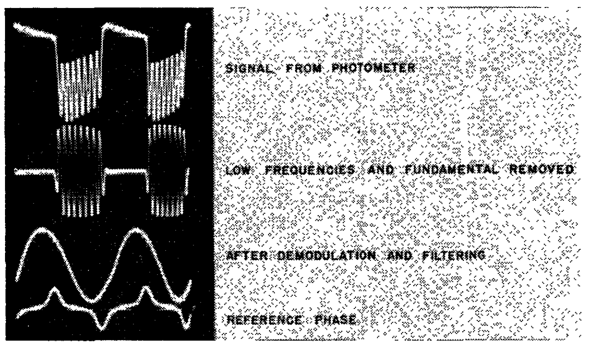

SIGNAL FROM PHOTOMETER
LOW FREQUENCIES AND FUNDAMENTAL REMOVED
AFTER DEMODULATION AND FILTERING
REFERENCE PHASE

Fig. 5—Typical oscilloscope presentation from simulator.

graphs of several background distributions and superimpose on them a more brilliant spot to simulate our function J_S. Let us choose a rotational scanning operation where we rotate the filter shown in Fig. 3.

The signal from this system is converted to an electrical signal by the photometer and we may observe the output on an oscilloscope (Fig. 5) and indeed be able to obtain a frequency spectrum by using a wave analyzer. We may, in this way, examine many sizes and intensities of spots and the effect of the different backgrounds on our output signal.

It is interesting to note a very special case where our spot is made very small and the background removed; this condition approaches the unidimensional case. A plot of the transform is shown in Fig. 6.

Conclusions

The analogy between the two-dimensional space filter problem and the one-dimensional electrical filter problem is seen to be formally complete. The necessity to obtain time dependent space filtering, for various practical applications, modifies the formal symmetry of the analogy without changing the basic two-dimensional

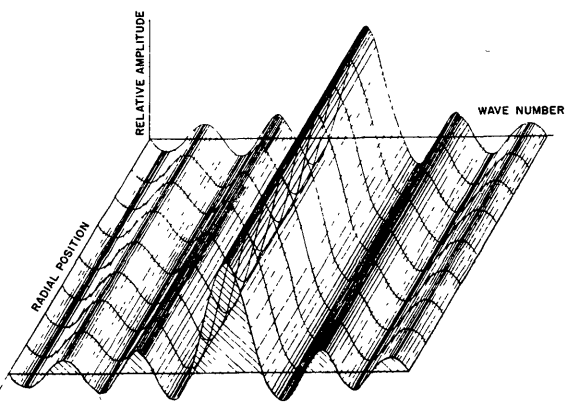


Fig. 6—Transform of n-wedged filter.

character. In many cases of electrical filter design, the analytic attack, though straightforward, proves to be less direct than an analog computation. Similarly, for the two-dimensional filter, an appropriate analog, together with our analytic understanding of the filter process, provides the designer with an effective solution for his problem.

Ideal Transformers in the Synthesis of Analog Computer Circuits

R. H. MacNEAL† and G. D. McCANN†

Introduction

THE NEED FOR ideal transformers in the solution of network synthesis problems, especially in multi-terminal problems, has long been recognized.[1–3] Nevertheless, there has been a general feeling that solutions to synthesis problems containing ideal transformers are of little more than academic interest because of the impracticability of constructing transformers good enough to be called ideal.[4] Whenever possible, inductances in the network are associated with the transformer so that it can be replaced by coils having self and mutual inductance.[2] This attitude toward the ideal transformer is inappropriate in at least one branch of electrical engineering where network synthesis techniques are employed, namely in analog computing of the direct analogy (or network analyzer) type.[5] Such computers customarily operate in the audio frequency range and contain high quality passive circuit elements which are adjustable in small steps. In the design of such computers the choice between "ideal" transformers and mutual inductance coils is an easy one to make. The availability of "supermalloy,"[6] which has an initial permeability of 70,000 or more makes

† Department of Electrical Engineering, California Institute of Technology, Pasadena, Calif.

[1] W. Cauer, "Ideale transformatoren und lineare transformationen," *Elec. Nach. Tech.*, vol. 9, p. 157; May, 1932.

[2] W. Cauer, Ein Reacktanztheorem; Sitz. d Preuss Akad. der Wissen; Phys. Math., pp. 673–681; 1931.

[3] C. M. Gewertz, "Network Synthesis," The Williams and Wilkins Co., Baltimore, Md.; 1933.

[4] R. Bott and R. J. Duffin, "Impedance synthesis without use of transformers," *Jour. Appl. Phys.*, vol. 20, p. 816; August, 1949.

[5] H. E. Criner, G. D. McCann, and C. E. Warren, "A new device for the solution of transient vibration problems by the method of electrical-mechanical analogy," *Jour. Appl. Mech.*, vol. 12, pp. 135–141; September, 1945

E. L. Harder and G. D. McCann, "A large scale general purpose electric analog computer," *Trans. AIEE*, Part I, vol. 67, pp. 664; 1948.

[6] R. M. Bozorth, "Ferromagnetism," D. Van Nostrand Co., New York, p. 143; 1951.

possible the design of ideal transformers which have parasite effects no more serious in practice than those of an ordinary inductor. The ideal transformer has in its favor a much greater versatility than the mutual inductance (making possible mutual resistance and capacitance) and probably a cost advantage as well, when the adjustability requirement is considered.

The use of ideal transformers in the design of analog computer circuits has been illustrated in a large number of papers, some of which are more than twenty years old.[7,8] Most of these papers are concerned with the derivation of an electrical analogy for a specific type of physical system (e.g., bending of a beam,[9] pin-ended truss,[7] airplane fuselage shell[10]). The approach of the present paper is somewhat more general than this, but still somewhat more specific than that of the network-synthesis theoretician. The field of interest is limited to analog computing, but the results are intended to apply to the synthesis of analogies for all kinds of mechanical systems.

It is recognized that information concerning a complex system may be presented in a variety of different ways. The information may come as a detailed description of the inner workings of the system, or this information may already have been combined with the aid of general physical principles to form equations of state. Another possibility is that the information may come in the form of measurements taken in tests made on the system, or in tests made on parts of the system. In the latter instance the parts may be installed, or they may be removed from the system when tested. In some cases the information may be presented in all of these ways at once.

In the papers cited above which are concerned with the derivation of electrical analogies for specific systems, it is assumed that the information is presented as a detailed description of the inner workings of the system. The usual (or best) method employed for deriving analogies when information is presented in this fashion is first to subdivide the system to its ultimate elements—springs, masses, joints, beam bending elements, etc.; then to write down analogies for these elements by inspection; and finally to interconnect the elements. When all or part of the information is presented in the form of equations of state or as the result of tests, this method is obviously impracticable for the parts of the system affected. Under such circumstances the system, or its affected parts, must be regarded as mechanical black boxes into which we are not permitted to look but which must be replaced with equivalent electrical black boxes which behave in an identical (or rigorously analogous) manner at their terminals. Unfortunately from the point of view of the computer engineer, the number of such terminals is large and the methods for deriving the networks are not altogether straightforward.

The scope of this paper is restricted to the synthesis of systems characterized by ordinary algebraic equations with constant, real coefficients. For such systems general methods of synthesis are well known.[1]

The All-Resistor Network

The need for ideal transformers in network synthesis problems can be illustrated by considering the limitations placed on the form of the equations of a network not containing transformers, or, in view of the restriction to equations with real coefficients, a network of resistors. We shall consider that the unknowns appearing in the equations of a network with $n+1$ nodes are the voltages from each of n nodes to a single reference node. The form of the equations written in matrix form is

$$[Y][E] = [I] \tag{1a}$$

or

$$\begin{bmatrix} Y_{11} & Y_{12} \cdots Y_{1n} \\ Y_{21} & Y_{22} \cdots Y_{2n} \\ \cdot\ \cdot\ \cdot & \cdot\ \cdot\ \cdot\ \cdot \\ Y_{n1} & Y_{n2} \cdots Y_{nn} \end{bmatrix} \begin{bmatrix} E_1 \\ E_2 \\ \cdot \\ E_n \end{bmatrix} = \begin{bmatrix} I_1 \\ I_2 \\ \cdot \\ I_n \end{bmatrix}. \tag{1b}$$

A network satisfying these equations for the case $n=3$ is shown in Fig. 1. Values of conductance of the

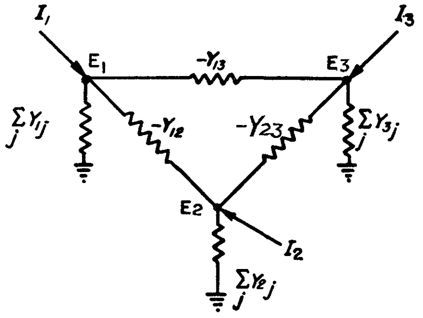


Fig. 1—All-resistor network.

resistors calculated by elementary network analysis are shown in the figure. The network in Fig. 1 is the most general all-resistor network with three independent node pairs. If all of the resistors in this network are to be *positive* and *bilateral*, then the following restrictions must be placed on the coefficients in (1b).

$$\begin{aligned} &Y_{ij} = Y_{ji} \leqq 0 && i \neq j \;\; i, j = 1 \cdots n \\ &\sum_{j=1}^{n} Y_{ij} \geqq 0 && i = 1, 2 \cdots n. \end{aligned} \tag{2}$$

These conditions are far more restrictive than the general conditions for synthesis with passive elements (including transformers), as will be seen.

[7] V. Bush, "Structural analysis by electric circuit analogies," *Jour. Franklin Inst.*, vol. 217, pp. 289–329; March, 1934.

[8] R. R. M. Mallock, "An electrical calculating machine," *Proc. Roy. Soc.*, ser. A, vol. 140, pp. 457–483; 1933.

[9] G. D. McCann and R. H. MacNeal, "Beam vibration analysis with the electric analog computer, *Jour. Appl. Mech.* vol. 72, pp. 13–26; March, 1950.

[10] R. H. MacNeal, "Electrical Analogies for Stiffened Shells with Flexible Rings," NACA TN 3280; December, 1954.

One way of easing the above restrictions is to transform the unknown variables by a linear co-ordinate transformation. Such a transformation might be a simple scale factor multiplication (a diagonal transformation) or it might involve the reduction of $[Y]$ to the unit matrix, i.e., a solution of the problem giving $[E]$ explicitly in terms of $[I]$. Such transformations are not generally permissible in theoretical network synthesis, nor are they permissible from the point of view of analog computing. The terminal voltages $[E]$ and the input currents $[I]$ must be represented in the network. The practical reason for this is that the system represented by (1) may well be only a part of a larger system, in which case $[E]$ and $[I]$ represent also the currents and voltages in other parts of the system.

The All-Transformer Network

The equations of a three-winding ideal transformer, shown in Fig. 2, are

$$\frac{E_1}{N_1} = \frac{E_2}{N_2} = \frac{E_3}{N_3}, \tag{3}$$

$$N_1I_1 + N_2I_2 + N_3I_3 = 0. \tag{4}$$

Stated in words, the general conditions for an ideal, multiwinding, single-core transformer are that the voltages across the various windings are directly proportional to their turns ratios and that the sum of the ampere turns (magnetomotive force) is zero.

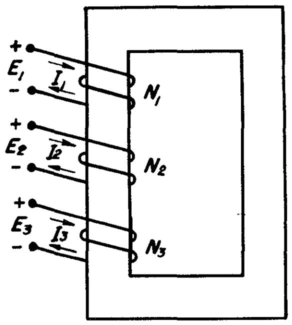


Fig. 2—Ideal transformer.

Consider now the network of transformer windings shown in Fig. 3. All the coils in one horizontal row are wound on the same core of an ideal transformer. Coils in different rows are not magnetically coupled. The relative number of turns in each winding is indicated by the symbol T_{ij}. From the manner of connection and (3), the voltage E_1 is given by

$$E_1 = T_{11}\bar{E}_1 + T_{12}\bar{E}_2 + T_{13}\bar{E}_3 \tag{5}$$

or, in general,

$$[E] = [T][\bar{E}]. \tag{6}$$

Also, from (4)

$$\bar{I}_1 = T_{11}I_1 + T_{21}I_2 + T_{31}I_3 \tag{7}$$

or, in general,

$$[\bar{I}] = [T]^T[I]. \tag{8}$$

(In this equation, the superscript t indicates that the matrix $[T]$ has been transposed.)

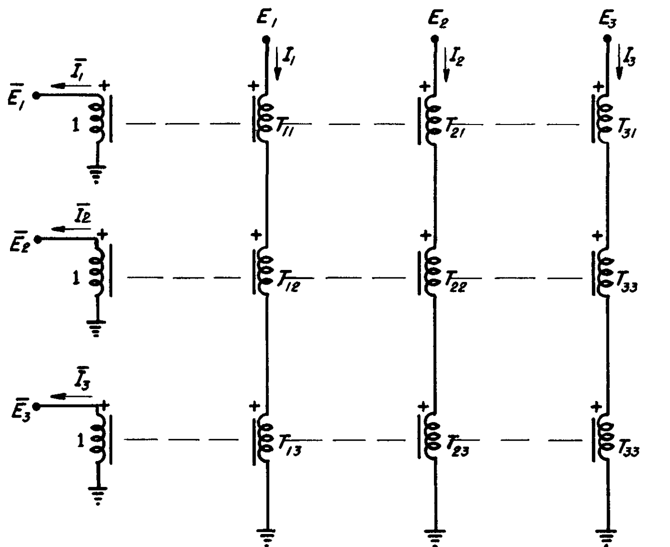


Fig. 3—All-transformer network.

Eqs. (6) and (8) can be regarded as a *workless* transformation from variables $[\bar{I}]$, $[\bar{E}]$ to variables $[I]$, $[E]$, since

$$[I]^T[E] = [I]^T[T][\bar{E}] = \{[T]^T[I]\}^T[\bar{E}]$$
$$= [\bar{I}]^T[\bar{E}]. \tag{9}$$

Hildebrand gives an explanation of the matrix identity employed in (9).[11] Note that from the point of view of physical realizability the only restriction on the elements of $[T]$ is that they be real numbers. Hence the network of Fig. 3 is capable of solving (6) with arbitrary real coefficients (if $[T]$ is nonsingular). Voltages $[E]$ are established by external generators and the voltages $[\bar{E}]$ read off. A practical version of this network computer was invented by Mallock[8] in 1933 and sold commercially. Considering the improvement in transformer iron in the last twenty years this network has inviting characteristics as a computer even today.

From the point of view of network synthesis a more important characteristic of this network is that it provides an *electrical* means for carrying out a co-ordinate transformation. Thus the objection to co-ordinate transformation mentioned in the previous section is overcome because *both* the original and the transformed coordinates appear in the network. Let us apply the transformation given by (6) and (8) to (1a):

$$[Y][T][\bar{E}] = [I]$$
$$[T]^T[Y][T][\bar{E}] = [\bar{I}]. \tag{10}$$

[11] F. B. Hildebrand, "Methods of Applied Mathematics," Prentice-Hall, Inc., New York, p. 13; 1952.

Define $[\bar{Y}]=[T]^T[Y][T]$. Eq. (10) can be synthesized by inspection if a transformation can be found which reduces $[\bar{Y}]$ to a diagonal matrix with either zero or positive elements on the principal diagonal and zero elsewhere. If such a transformation exists, the original admittance matrix $[Y]$ is said to be *positive semi-definite*. Hence a sufficient condition for the passive synthesis of (1a) is that the matrix $[Y]$ be positive semi-definite. An efficient transformer network for carrying out the synthesis is described in the next section.

The co-ordinate changing properties of transformer networks have been used extensively in connection with the direct analogy type of analog computer.[12,13] For example, they provide the means for interconnecting analogies for parts of a mechanical system which are oriented in different space directions (e.g., the interconnection of an airplane's fuselage and its swept-back wings). Fig. 4 shows a very simple transformer circuit for synthesizing a negative coupling element.

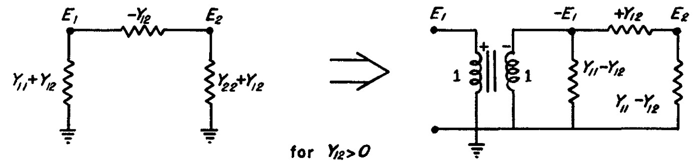


Fig. 4—Synthesis of negative coupling element.

Cauer Networks

Cauer's network, as originally presented,[1] synthesizes a system with equations presented in the form

$$[Z][I]=[E]. \tag{11}$$

The resulting equation in terms of transformed variables is

$$[\bar{E}]=[R][\bar{I}], \tag{13}$$

where it is assumed that

$$[R]=\begin{bmatrix} R_{11} & 0 & 0 & \cdots 0 \\ 0 & R_{22} & 0 & \cdots 0 \\ 0 & 0 & R_{33} & \cdots 0 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdots R_{nn} \end{bmatrix} \tag{14}$$

and $R_{ii}\geqq 0$.

(If any R_{ii} equal zero it is assumed that they occupy the last positions in the matrix. This condition can be satisfied by rearranging rows and columns in the T matrix.)

Premultiply (13) by $[T]$:

$$[E]=[T][\bar{E}]=[T][R][\bar{I}]=[T][R][T]^T[I]. \tag{15}$$

Hence

$$[Z]=[T][R][T]^T. \tag{16}$$

The problem is to find matrices $[T]$ and $[R]$ which satisfy this equation. Let $[T]$ be a *triangular* matrix of the form

$$[T]=\begin{bmatrix} 1 & 0 & 0 & \cdots 0 \\ T_{21} & 1 & 0 & \cdots 0 \\ T_{31} & T_{32} & 1 & \cdots 0 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ T_{n1} & T_{n2} & T_{n3} & \cdots 1 \end{bmatrix}. \tag{17}$$

Then

$$[T][R][T]^T=\left[\begin{array}{c|c|c|c} R_{11} & R_{11}T_{21} & R_{11}T_{31} & \cdots \\ \hline R_{11}T_{21} & R_{22}+R_{11}T_{21}^2 & R_{11}T_{21}T_{31}+R_{22}T_{32} & \cdots \\ \hline R_{11}T_{31} & R_{11}T_{21}T_{31}+R_{22}T_{32} & R_{33}+R_{11}T_{31}^2+R_{22}T_{32}^2 & \cdots \\ \hline \cdots & \cdots & \cdots & \cdots \end{array}\right]. \tag{18}$$

In this section Cauer's network will first be derived in its original form and then in a form suitable for the synthesis of (1a). It is assumed that the matrices $[Z]$ and $[Y]$ are positive semi-definite. This does not imply that they are necessarily nonsingular.

As stated in the previous section, the network can be synthesized if a co-ordinate transformation is found which transforms $[Z]$ into a diagonal matrix.

Let this transformation be given by

$$[E]=[T][\bar{E}]$$
$$[\bar{I}]=[T]^T[I]. \tag{12}$$

[12] W. T. Russell, "Lumped Parameter Analogies for Continuous Mechanical Systems," Calif. Inst. Tech., Ph.D. thesis; 1950.

[13] R. H. MacNeal, G. D. McCann, and C. H. Wilts, "The solution of aeroelastic problems by means of electrical analogies," *Jour. Aero. Sci.*, vol. 18, pp. 777–789; December, 1951.

Note that, given the elements of $[Z]$, the elements of $[T]$ and of $[R]$ can be evaluated *one at a time*. Explicitly,

$$R_{11}=Z_{11};\quad T_{21}=Z_{12}/R_{11};\quad T_{31}=Z_{13}/R_{11}$$
$$R_{22}=Z_{22}-R_{11}T_{21}^2,\ \text{etc.} \tag{19}$$

The triangular transformation matrix is efficient from the point of view of electrical synthesis since fewer windings are required than in the case of a full transformation matrix. Cauer's "Z-network" is shown in Fig. 5(a). The barred co-ordinates can be eliminated by placing the R's on the other side of the windings, as shown in Fig. 5(b). The total number of windings required in the synthesis of n simultaneous equations by Cauer's network is $n\cdot(n+1)/2-1$. If multiwinding transformers are not available, they may be replaced by $n\cdot(n-1)/2$ 2-winding transformers.

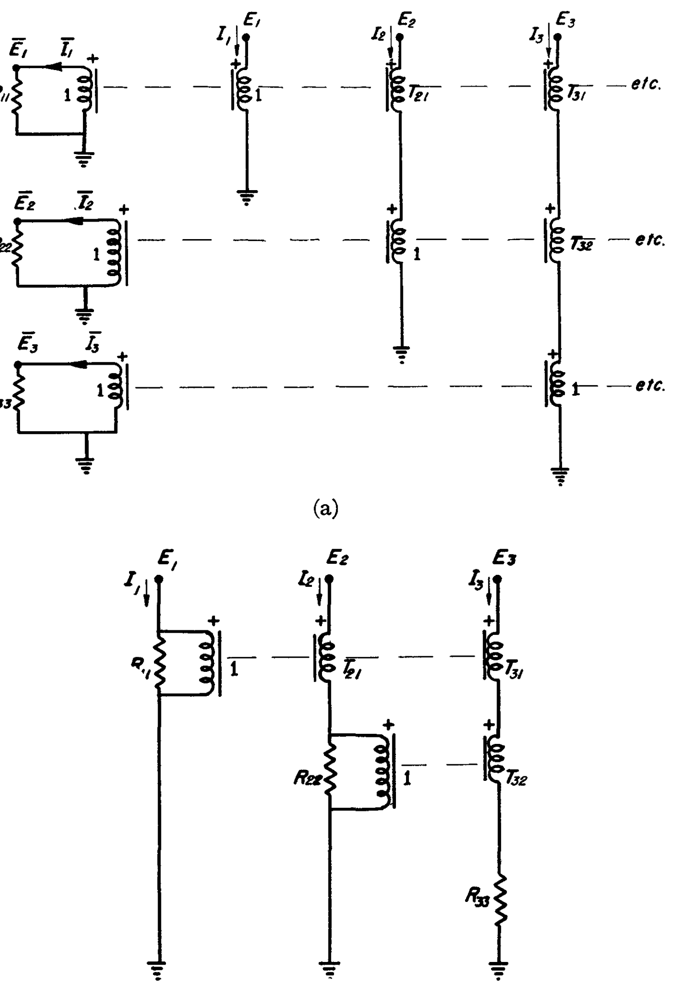


Fig. 5—Cauer's Z-network, (a) with barred co-ordinates represented, (b) with barred co-ordinates eliminated.

The necessary and sufficient conditions for the synthesis of (11) by Cauer's network may be stated in a way that does not involve matrix theory. If (11) is to represent a passive electrical network then it is evidently necessary that the input impedance at any given terminal must be positive (or at least zero) whether some, none, or all of the other terminals are short-circuited. It is easy to demonstrate that this condition is also a sufficient one for the synthesis of Cauer's network. Every element in Cauer's network can be computed by the formulas in (19) so that the only real question is whether or not the resistors are positive. From inspection of Fig. 5(b) it is evident that R_{11} is the input impedance at terminal 1 with all other terminals open-circuited; R_{22} is the input impedance at terminal 2 with terminal 1 short-circuited and all other terminals open-circuited; R_{33} is the input impedance at terminal 3 with terminals 1 and 2 short-circuited, and all others open-circuited; etc. Hence all resistors are positive (or zero).

We will next construct Cauer's "Y-network" for a system specified by equations of the form of (1a). This network, shown in Fig. 6, is the "dual" of the "Z-network" shown in Fig. 5. The analysis proceeds by analogy with the derivation of the Z-network. Let the transformation be

$$[I] = [S][\bar{I}]$$
$$[\bar{E}] = [S]^T[E]. \qquad (20)$$

Then if

$$[\bar{I}] = [G][\bar{E}] \qquad (21)$$

where $[G]$ is a diagonal matrix, it is evident that

$$[Y] = [S][G][S]^T. \qquad (22)$$

$[S]$ is chosen as a triangular matrix identical in form to $[T]$ so that the elements of $[S]$ and $[G]$ are determined by formulas rigorously analogous to (19).

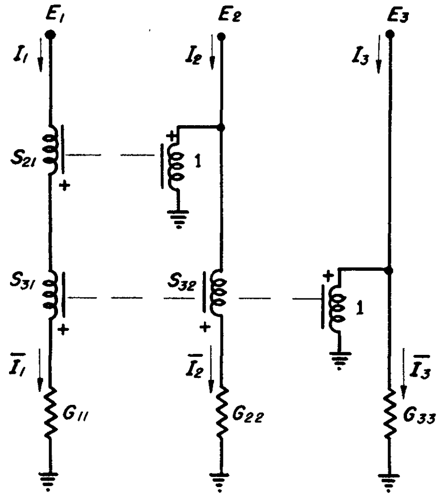


Fig. 6—Cauer's Y-network.

Cauer's networks are important because of their generality. Unfortunately for practical applications, the number of transformer coils required increases approximately as the square of the number of terminals. On the other hand, under special conditions no transformer at all may be required (the all-resistor network). An important unsolved problem is the determination of the *minimum* number of transformers required to synthesize a given system of equations and the construction of a network exhibiting this minimum number.[14]

Synthesis of Systems with "Restrained" Terminals

Many of the problems that arise in analog computing can be translated into problems in abstract network synthesis, but sometimes the problem is one that would not independently occur to a network theoretician as an interesting and useful problem on which to work. Such is the nature of the problem considered in this section. As has frequently been demonstrated, the abstract mathematical sciences depend for their growth on an influx of practical problems, however much these prob-

[14] J. K. Delson, "Networks Involving Ideal Transformers," Calif. Inst. Tech., Ph.D. thesis; 1953.

lems may have been predigested and stripped of their practical aspects.

We may assume, without loss of generality, that electrical analogies for mechanical systems are always set up with force analogous to current and displacement analogous to voltage. Equations written in the form of (1a) are then force equations for the mechanical system and are equations of state (Newton's equations, Lagrange's equations) in the usual sense. Mechanical equations written in the form of (11) ordinarily describe the results of tests, i.e., the displacements that are produced by application of a set of external forces on a body. Hence both forms of Cauer's network are useful in analog computing. The Y-network is useful when information concerning the system is presented as equations of state, and the Z-network is useful when information concerning the system is presented as a matrix of "influence coefficients." In the latter case the information so presented never describes the entire system, for if it did it would constitute a *solution* of the problem. Hence, the influence coefficients describe tests made on a subsystem while it is disconnected from the rest of the system. We can regard the subsystem as a black box electrical network with $n+1$ terminals (one ground terminal). The test the results of which are described by (11) consists of the insertion of a current at each node in turn with all other nodes free (open-circuited), and the measurement of the resulting terminal voltages. It should be recognized that, from the mechanical point of view, this might be a *meaningless* experiment. As an elementary example consider a straight segment of rod capable of resisting tension and compression. In the analogous electrical circuit the terminal voltages are the axial displacements at the ends. (See Fig. 7.) Such a

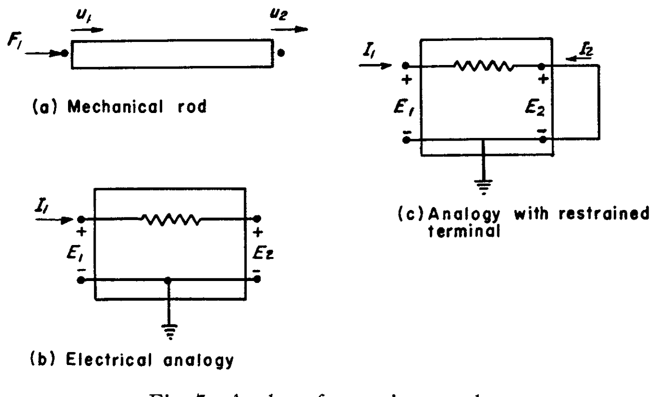


Fig. 7—Analogy for tension member.

system cannot be in equilibrium under the action of a force exerted at one end of the rod. In this example the $[Z]$ matrix does not exist, a fact which may variously be regarded as due to the singularity of the corresponding $[Y]$ matrix; as due to the existence of a rigid body translational degree of freedom; or, electrically, as due to the fact that no connection with ground is provided. The general elastic body has 6 elastic degrees of freedom (3 translations and 3 rotations) corresponding to a $[Y]$ matrix whose determinant and all of whose minor determinants of the first 5 orders vanish (defect = 6). An obvious way to remedy the situation is to *restrain* enough terminals to prevent rigid body motion. If this is done the $[Z]$ matrix in (11) will not describe the entire subsystem but only the influence which the remaining *free* terminals have on each other. A method for obtaining a complete description of the partially restrained system that is suitable for network synthesis is the subject of this section.

A black box is shown in Fig. 8 with any number of terminals (five shown). These are divided into two

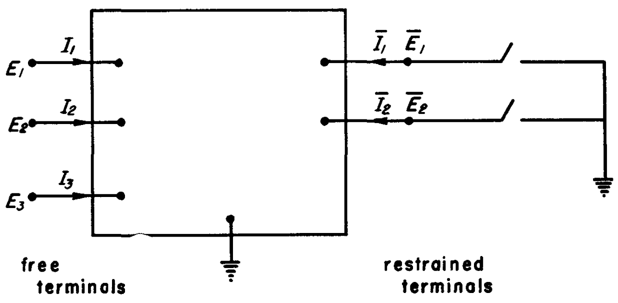


Fig. 8—System with some free terminals and some restrained terminals.

groups; free terminals (unbarred quantities), and restrained terminals (barred quantities). The restrained terminals are so chosen that rigid body motion is impossible when they are grounded. Their number may, however, exceed the minimum number required to prevent such motion. The $[Y]$ matrix for the black box exists and may be partitioned according to the free and restrained terminals.

$$\left[\begin{array}{c|c} Y_{ii} & Y_{ji} \\ \hline Y_{ij}^T & Y_{jj} \end{array}\right]\left[\begin{array}{c} E_i \\ \hline \bar{E}_j \end{array}\right]=\left[\begin{array}{c} I_i \\ \hline \bar{I}_j \end{array}\right], \tag{23}$$

where

$$i = 1, 2 \cdots p \qquad j = p+1, \cdots n.$$

In accordance with the above discussion $[Y_{ii}]$ is nonsingular. Its inverse is defined as $[Z]$, i.e.:

$$[Y_{ii}]^{-1} = [Z]. \tag{24}$$

We wish to rewrite (23) so that $[E_i]$, voltage at free terminals, and $[\bar{I}_j]$, current at restrained terminals, both appear on the right.

Write the top half of (23) separately and solve for $[E_i]$:

$$[Y_{ii}][E_i] + [Y_{ij}][\bar{E}_j] = [I_i] \tag{25}$$

$$[E_i] = [Z][I_i] - [Z][Y_{ij}][\bar{E}_j]. \tag{26}$$

Write the bottom half of (23) separately:

$$\begin{aligned}[\bar{I}_j] &= [Y_{ij}]^T[E_i] + [Y_{jj}][\bar{E}_j] \\ &= [Y_{ij}]^T[Z][I_i] + \{[Y_{jj}] - [Y_{ij}]^T[Z][Y_{ij}]\}[\bar{E}_j]. \end{aligned} \tag{27}$$

Define

$$[T] = -[Z][Y_{ij}]$$
$$[\bar{Y}] = [Y_{jj}] - [Y_{ij}]^T[Z][Y_{ij}]. \qquad (28)$$

Then, combining the results of the last three equations:

$$\left[\begin{array}{c|c} Z & T \\ \hline -T^T & \bar{Y} \end{array}\right]\left[\begin{array}{c} I_i \\ \hline \bar{E}_j \end{array}\right] = \left[\begin{array}{c} E_i \\ \hline \bar{I}_j \end{array}\right]. \qquad (29)$$

Each of the matrices appearing in (29) has a definite significance in terms of physical measurements on a mechanical system. $[Z]$ is the matrix of influence coefficients with the restrained terminals restrained. $[T]$ is a matrix giving the displacements at the free terminals due to displacements at the restrained terminals. If $[\bar{Y}]=0$, these displacements correspond to rigid body motion. $[\bar{Y}]$ is the matrix of force coefficients for the restrained terminals with the free terminals unrestrained. Hence if the number of restrained terminals is just equal to the minimum number required to prevent rigid body motion, $[\bar{Y}]$ will equal zero.

A network satisfying (29) is shown in block form in Fig. 9. This network employs a Cauer Z-network, a

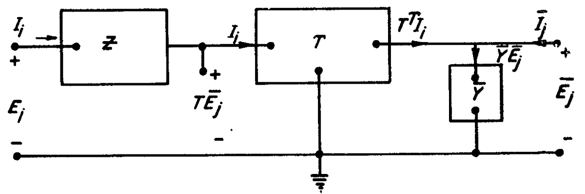


Fig. 9—General synthesis of system with restrained terminals.

Cauer Y-network, and an all-transformer network. The Cauer Z-network differs from that shown in Fig. 5(b) in that the terminals at the bottom of Fig. 5(b) are connected to the input of the T-network instead of being grounded. An explicit representation of the network for the case of three free and two restrained terminals is shown in Fig. 10. It is interesting to observe that the number of transformer windings and the number of resistors required in this network are respectively equal to the number that would be required in an all-Y or all-Z representation.

Practical Applications

The analog computer solution of problems in structural analysis furnishes many illustrations of the use of the networks described in this paper.

Consider an airplane wing, which may be represented as a cantilever beam with vertical bending and torsional displacements, connected to the airplane's fuselage. Information concerning the wing is presented as a detailed description of its structural properties (EI and GJ vs spanwise station). Information concerning the fuselage is contained in the statement that it may be considered to be rigidly restrained at its center of gravity (for the particular problem at hand) and in a series of measurements giving the vertical displacement and pitching and rolling rotations at the side of the fuselage as a result of load applied at this point. Electrically the airplane wing can be represented by a standard beam analogy,[9] while the fuselage can be represented by a Cauer Z-network, such as that shown in Fig. 5(b). If the statement that the fuselage is fixed at its center of gravity is now amended to read that the inertia loads of the fuselage may be considered to be concentrated at the center of gravity, the fuselage can be represented by a circuit similar to Fig. 10. In this case the Y-network is a set of

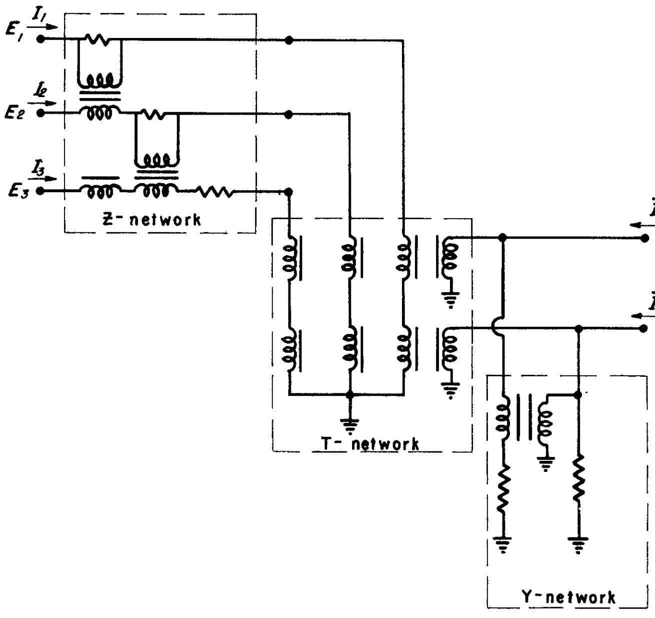


Fig. 10—Synthesis of system shown in Fig. 8.

three uncoupled capacitors representing the mass and the rolling and pitching moments of inertia of the fuselage.

An important type of analog computer circuit is the analogy for what may be termed the *one-dimensional element*. This is a structural element of narrow cross section for which the internal strains can be derived with sufficient accuracy from knowledge of the (approximately) rigid motions of its cross sections. Examples are straight, curved, and twisted beams with bending, torsional, and extensional degrees of freedom. Analogies can be synthesized for one-dimensional elements by first replacing distributed loads (if any) by concentrated loads and then deriving analogies for the segments between load points by methods described in this paper. Since the restraint of six degrees of freedom at one end of a one-dimensional element is just sufficient to prevent rigid body motion, the circuit will not contain a $\bar{y}$-network and the T-network can be derived from rigid body considerations.

As a subcase consider the straight segment of uniform beam shown in Fig. 11 for which a bending analogy is desired. In this case the $[T]$ matrix is given by

$$\begin{bmatrix} W \\ \theta \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} \bar{W} \\ \bar{\theta} \end{bmatrix} = \begin{bmatrix} 1 & l \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \bar{W} \\ \bar{\theta} \end{bmatrix}. \quad (30)$$

The $[Z]$ matrix (obtained most easily from strain energy and Castigliano's theorem), is

$$\begin{bmatrix} W \\ \\ \theta \end{bmatrix} = \begin{bmatrix} Z_{11} & Z_{12} \\ \\ Z_{21} & Z_{22} \end{bmatrix} \begin{bmatrix} V \\ \\ M \end{bmatrix} = \begin{bmatrix} \dfrac{l^3}{3EI} & \dfrac{l^2}{2EI} \\ \\ \dfrac{l^2}{2EI} & \dfrac{l}{EI} \end{bmatrix} \begin{bmatrix} V \\ \\ M \end{bmatrix}. \quad (31)$$

The $[Y]$ matrix is zero.

The resulting network is shown in Fig. 12. This form of the electrical beam analogy was first derived by Russell.[12]

In another paper the application of ideal transformers to the synthesis of purely reactive networks will be described and their importance for analog computer methods of vibration analysis will be pointed out.

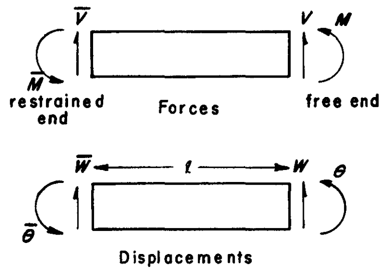


Fig. 11—Beam bending element.

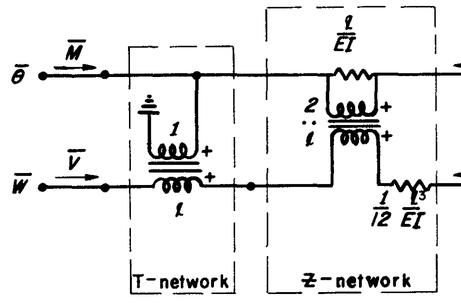


Fig. 12—Synthesis of analogy for beam bending element.

A New Approach to Grounding in DC Analog Computers*

C. M. EDWARDS†

Summary—**The difficulty of minimizing ground current effects and the problems caused by offset voltages in the development of dc analog computers is presented. A conventional grounding system is discussed and a ground isolation system is disclosed as a means of greatly reducing the existing differences in chopper references.**

Introduction

In THE PAST, considerable attention has been given to two major problems in the development of dc electronic analog computers. These problems are the difficulty of minimizing drift in associated electronic circuits and the difficulty of minimizing ground current effects.

Drift in the electronic circuits is, in general, caused by grid current and cathode potential changes in the low level stages of the dc amplifiers. Unless extreme care is taken in the design of a conventional dc amplifier, over a short time interval the drift referred to the input stage will be in the range of 10 to 100 millivolts. Thus a dc computing amplifier, operating as an integrator with a full scale output of 100 volts and a gain factor of one second, might drift 0.1 per cent of full scale in one second if the drift referred to the input is 100 millivolts.

Since a drift of this magnitude can hardly be tolerated in a complex computer, considerable effort has been expended to minimize dc amplifier drift. The most successful technique developed to date[1] involves the use of a stabilizing circuit which provides low-frequency, driftless gain ahead of the drift point. A dc computing amplifier, so stabilized, can be readily maintained to have less than 0.5 millivolt drift referred to the input.

Ground current effects in a large scale dc analog computer can be troublesome from a number of standpoints. The most significant effects are those associated with the generation of offset voltages and the determination of system stability. Although the problem of stability should not be minimized, it does not present as great a problem in dc analog computers restricted to the audio frequency range as does the problem of offset voltages.

Offset voltages cause computing errors in much the same way as amplifier drift causes errors. The ground current offset problem has been greatly accentuated by the advent of the stabilized amplifier with its inherent low drift. If it is important to maintain amplifier drift under 0.5 millivolt, then it is equally important to reduce offset voltages to under 0.5 millivolt. Such a re-

* The system described in this article was developed in connection with a program sponsored by Special Devices Center, Office of Naval Research, Department of the Navy through the Bureau of Aeronautics.

† Res. Labs., Bendix Aviation Corp., Detroit, Mich.

[1] E. A. Goldberg, "Stabilization of wide-band direct current amplifiers for zero and gain," *RCA Rev.*, vol. II, pp. 296–300; June, 1950.

quirement is difficult to meet by conventional "brute force" methods which involve the use of large quantities of copper to obtain low impedance grounds and corresponding low voltage drops.

The Offset Problem

Cathode currents and signal currents flowing in common ground circuits are the principal sources of offset voltages. In general, cathode current may be considered as steady and can, therefore, be balanced out. Signal currents, however, are variable and difficult to compensate. In either case, balancing is a difficult job which rapidly becomes impossible as the complexity of the computer increases.

In considering the offset problem in a large computer installation (Fig. 1) the size of the ground copper bus in a rack may be estimated. This estimate is based on the requirement that the dc voltage drop from top to bottom be less than 0.1 millivolt with each rack containing as many as 40 amplifiers and each amplifier requiring, on the average, 0.06 ampere of B^+ cathode current. For seven feet, the average resistance should therefore be less than 0.012×10^{-3} ohms per foot or the equivalent of four number 4-0 cables. The number of cables is reduced to two if the B^- cathode current is considered to be approximately half the B^+ current.

Since, in terms of interconnecting the racks, the amount of copper required goes up proportionally, another approach appears to be in order.

Conventional Grounding System

If the grounding system which is conventional for interconnecting dc analog elements (indicated in Fig. 2) is considered, there is one ground established at each active component such as amplifiers 1, 2, and 3. Assuming that these amplifiers are mounted in the same rack, the respective cathode currents flow back to the primary supply through a common ground bus. By this same bus, the chopper ground reference points are established at each amplifier. Although the chopper currents (I_c) are themselves negligible, the chopper ground references are different by the products of the appropriate cathode currents (I_k) and bus impedance. The voltage differences among the chopper references and between these references and the established ground are represented by E_{k_1}, E_{k_2}, and E_{k_3}. Therefore, although the input E_1 to amplifier 1 may be zero, the input to both amplifiers 2 and 3 will not be zero because the respective chopper references are different and because the potentiometer (R_a) return is different from ground by the amount of E_{s_2}. Even in the conventional system, voltage E_{s_2} might have been larger if the separate return had not been established for the signal currents (I_s).

Voltages E_{a_1}, E_{a_2}, E_{b_1}, E_{b_2}, E_{c_1}, and E_{c_2} result from the impedance of the leads to the amplifier loads (R_a, R_b and R_c) when signal currents are present. These lead impedances, which are usually negligible, may be considered as errors in the impedance of the respective loads and voltage E_{a_3} may be considered as an error in

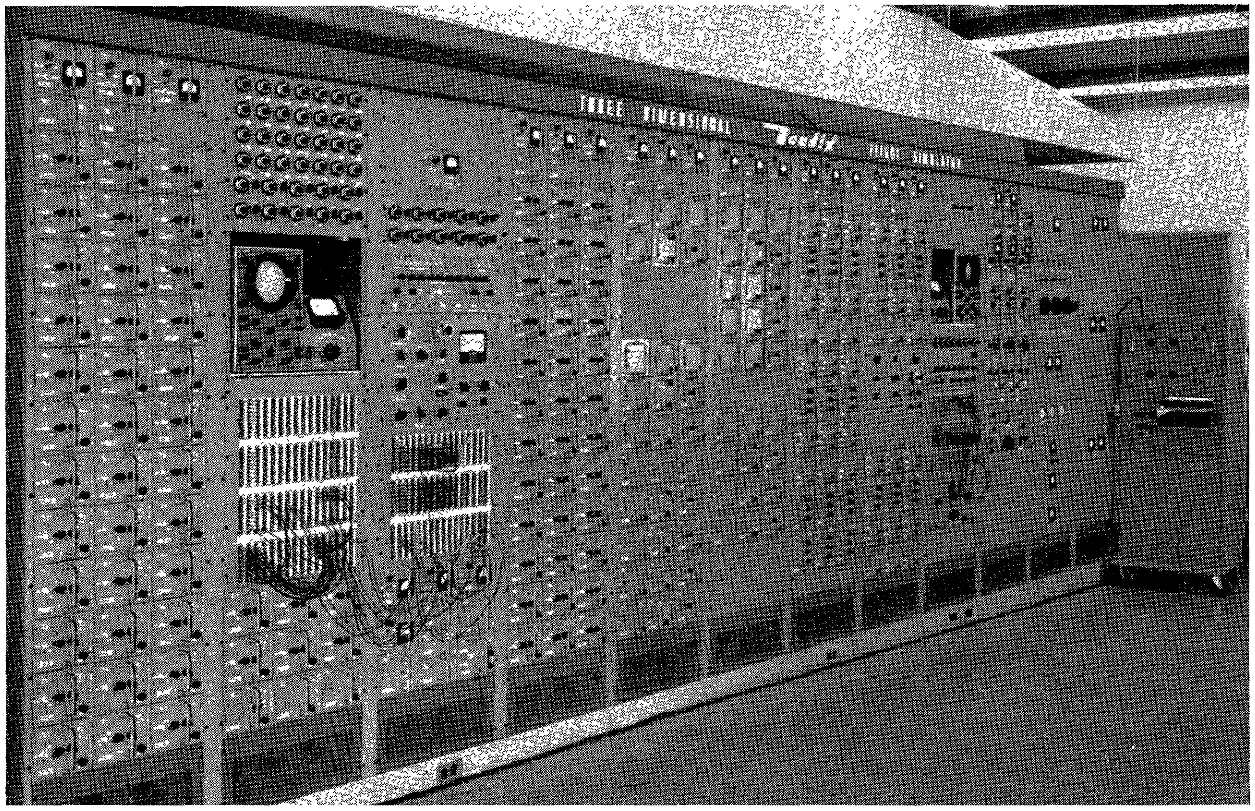


Fig. 1—Complex dc analog computer utilizing ground isolation system.

the voltage applied to amplifier 2. This voltage error is usually negligible also.

Because the computer has chopper stabilized amplifiers, it is possible to consider isolating the chopper grounds to eliminate the differences among the references and, therefore, the errors which result from the common bus system illustrated in Fig. 2.

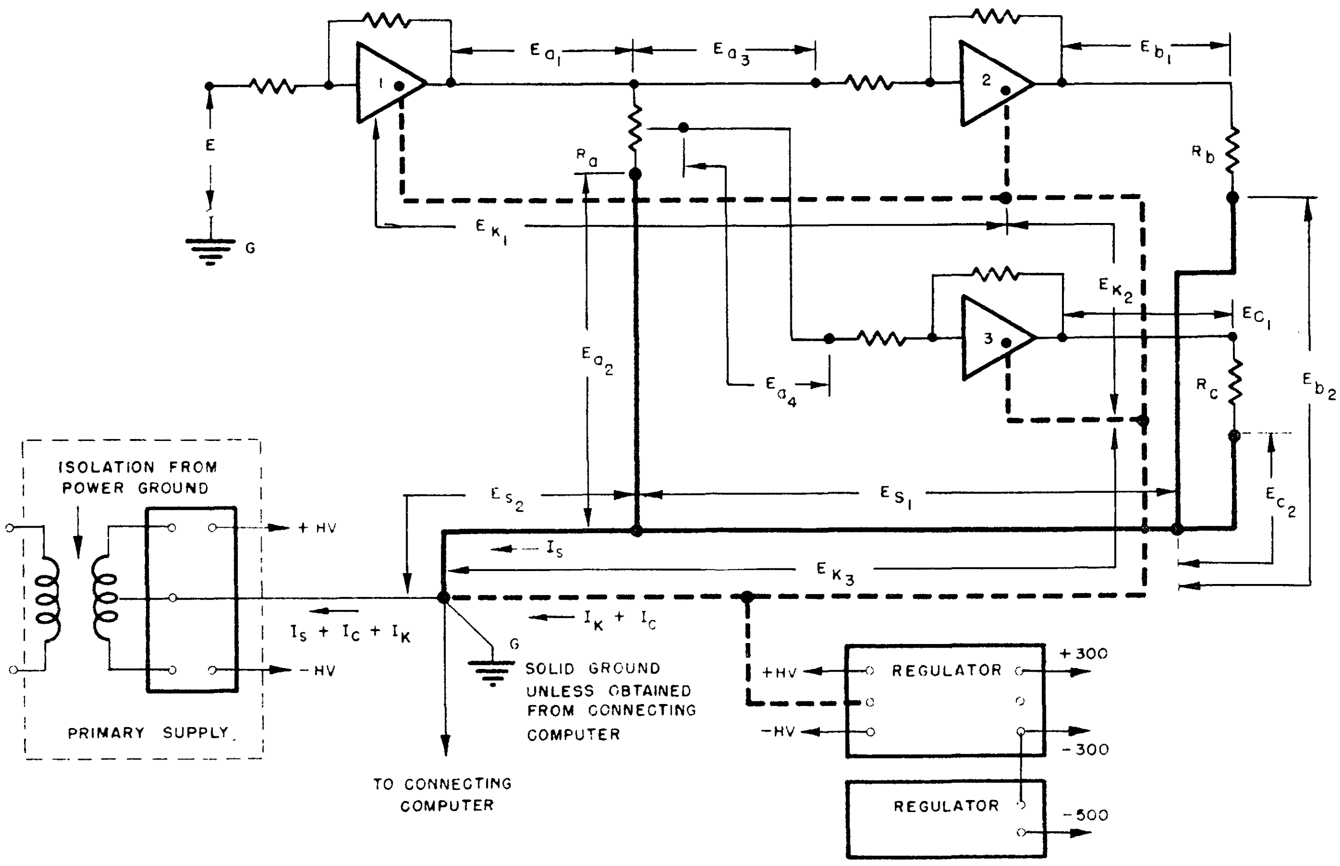


Fig. 2—Conventional ground system.

remains as the only principal source of ground voltage which couples two or more elements.

The importance of this effect is related to the position of the potentiometer (R_a) in the particular example illustrated in Fig. 3. Since it is common practice to locate all potentiometers in the same rack, it is possible to reduce these remaining coupling voltages to negligible proportions. This is accomplished by running individual leads from each potentiometer to the established ground in the rack near the group of potentiometers.

It should be realized that this approach to the grounding problem has been concerned with ground currents principally as they affect drift and offset, which are both low-frequency phenomena in a dc analog computer. At higher frequencies, above a few cycles per second, the chopper ground no longer establishes the reference point in the amplifier, so that the cathode current and the signal current buses become the principal reference points. In this higher frequency range, the effect of a few millivolts of coupling will be significant only as error (unwanted signal in terms of full scale) and as a source of system instability.

Chopper Isolation System

The ground isolation system which was developed is shown in Fig. 3. Experience to date has proved that the system virtually eliminates the differences among chopper references and, at the same time, it permits a reduction in the amount of copper required in the ground bus system. Essentially, three ground buses are provided back to the established ground point. Only chopper current (I_c) is flowing in the copper reference bus. Thus, the voltage drops are negligible and the differences among the choppers are considerably less than 0.1 millivolt.

The drop experienced in the cathode return bus because of current I_k may be ignored from a signal standpoint just as the drops in the regulated voltage buses have been ignored, so that the principal consideration for both is now impedance coupling as it affects system stability and crosstalk.

After development of the chopper isolation system, the common drop experienced in the signal return lines

Conclusion

The proposed ground isolation system virtually eliminates the differences among chopper references and permits a reduction in the amount of copper required in

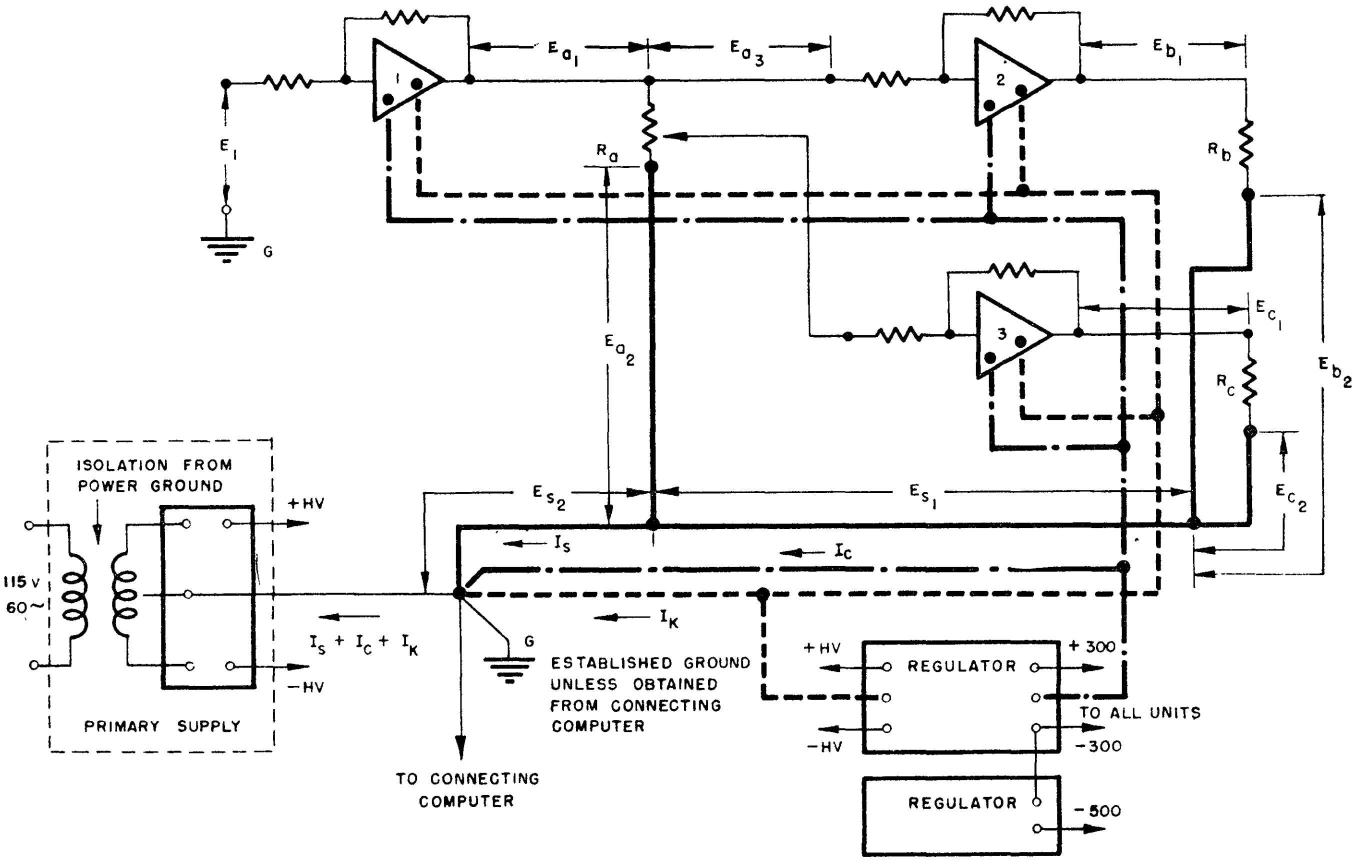


Fig. 3—Chopper ground isolation system.

the ground bus systems of complex dc analog computers. This approach to the grounding problem principally concerns ground currents as they affect low-frequency phenomena since, at higher frequency ranges, the effect of small coupling voltages is significant only as error and as a source of system instability.

The Need for Integration of Accounting Systems and the Design of Electronic Data-Processing Systems

PAUL KIRCHER[†]

A SHORT TIME ago Henry Dreyfuss, the man who designed your telephone and many other commercial products, gave an address on the principles of good industrial design. Mr. Dreyfuss emphasized his fundamental point by quoting an old Greek philosopher, who said that "the measure of all things is man."

No design can claim to be good unless it recognizes the needs, the abilities, the prejudices and even the hopes of the man who will use it. No matter how technically correct the design may be, if the man is not considered there will be friction wherever and whenever the man and the designed product come in contact.

This touchstone seems especially suited for the designer of an accounting system, for the designer of electronic data-processing equipment, and for the business manager who must try to integrate both of these with his system of planning and control.

Probably few of you would dispute the statement that too often there is friction between the accounting system and the operating man. Late reports, inaccurate reports, misunderstandings as to the reason for variances, arguments over cost allocations, and so on, are all too prevalent.

I might go further, and ask each of you—please raise your hand if you have never had any friction with your budget. . . .

[†] University of California, Los Angeles, Calif.

The inadequacies and expense of present methods of handling accounting data have opened a golden door for electronic data-processing systems. Opened it so wide, in fact, that for a while it seemed as though all the electronics engineer had to do was to walk right in. Engineers and mathematicians had developed computers which made calculations at lightning speeds. The machines were versatile, too. Just how versatile many of the builders thought them to be is indicated by the experience of a business acquaintance of mine. He asked a manufacturer's representative what his new computer could do. The answer was, word for word, "It can do anything!"

Perhaps I should hasten to add that this happened in San Francisco. No one in Los Angeles would ever make a claim like that. I think.

Well, as we all know, there have been some obstacles. The early scientific computers undoubtedly were a good start. But many additional features were necessary before the machines began to appear suitable for accounting.

Trend of Developments

It is interesting and instructive to trace the trend of these developments, which were designed to decrease the friction between accountants and the electronic data-processing machines. For example, the first computers were entirely numeric. It is possible to code words to numbers; I might even pass out a code list, and give this talk entirely in numbers. But the friction, obviously, would be terrific. So general-purpose business machines have become alpha-numeric.

Another example is input-output. I know a number of accounting systems men, eager to find a machine suited to their needs, who, in their search for equipment, have visited many manufacturers. In a number of cases I have accompanied them and watched their faces as they walked through the door of a manufacturer's display room. As soon as the accountants saw that the machine demonstration was going to use an electric typewriter for input, and for output, for all practical purposes the demonstration was over.

We are fortunate this morning to have several talks by men whose organizations have recognized this problem and are doing something about it.

Still other examples of the way in which electronic designers are improving their equipment, to make the machines more suitable for accounting, are the introduction of such things as variable word lengths, sorting routines, buffers at input and output, increased high-speed random access storage, built-in checking devices, and an increasing variety of commands.

Meanwhile, there is also a trend to be observed among accountants. For one thing, they have made a real effort to become acquainted with the abilities of electronic machines. If any of you have had the experience of trying to interest someone in a new product, you will recognize how important and helpful such an effort can be.

Accountants in many companies are moving to make their systems more compatible with the abilities of electronic equipment. They have studied the possibility of standardizing their code systems, and their reporting procedures, and the forms they use. They have assisted in educating the operating and staff personnel as to the significance of electronics in their companies.

Perhaps the best known of such efforts is the "common language" project. While one might have some reservations as to whether this approach does not overemphasize certain mechanical possibilities, it is evidence of real effort to come to grips with the problems.

Pioneer Experience

We have now reached a stage where it is possible to learn something from the experiences of the pioneers in the field of electronics in business. It is almost inevitable that a pioneer will do things differently than he would if he had more experience. No one is to blame for this; we are only to blame if we do not learn.

So, with no intent to be critical of the pioneers, but only to achieve better methods, let us examine some of these experiences.

Some companies have undertaken to install computers, but have insisted that the electronic system be used as a sort of service center. Operating managers are assured that nothing but the data processing will be changed. This we might call the "Safe But Sorry" approach.

Savings can be effected by using electronic equipment. However, in some cases where this approach has been used, the savings have not been as substantial as at first anticipated. More important, the great abilities of computers have been so restricted that their true potential is far from realized.

Too much caution is never going to eliminate the causes of friction which led businessmen to investigate the possibilities of electronics in the first place.

On the other hand, we have seen some groups who are going all-out. They have chosen their toughest problems to put on the computers. I call this approach "Let's Climb Mt. Everest."

Apparently this attitude has sometimes been adopted in the belief that it is necessary to prove the abilities of the machines, to make good the statement that a general-purpose computer can do everything. Or, in some cases, it may be the drive to achieve immense savings immediately.

Anyhow, this approach led one company to start with a routine which, according to reports, required the rather horrifying total of 240,000 program steps. Many exceptions existed. Since the high-speed memory of their machine could hold but a portion of such a program, many passes of the data would be required. As a result, the company has announced its decision not to use the computer on some of the exceptions.

We have often been assured that the ability to handle exceptions is one of the major advances that computers will bring over former procedures and equipments.

Here is a real source of friction. In the future, we can hope that companies will give more attention to the question whether plans calling for special computations are economically justified.

At the same time, it offers a challenge to the electronics engineer. Computers still are not flexible enough to handle anything and everything economically. I am not referring to some fantastic routine. I am speaking of a routine that was handled by other means before the computer was introduced, and again is being handled by these other means.

Then we have the companies who have ordered computers because it's fashionable. This approach might be called, "Look Ma, I'm Dancing!"

Several of these companies have found themselves in the embarrassing position of having to ask the manufacturers to postpone delivery of their machines. The companies had not made the necessary systems studies, prepared programs, and attended to all the other details that are necessary for a successful installation.

Still other companies have installed special-purpose computers and have operated them successfully on certain specific routines. But the companies have made little or no effort to extend the use of the equipment beyond these limited applications, or to explore the possibility of using different machines. I call this approach, "Jack, the Rabbit-Killer."

It seems a shame to see a big hundred-thousand-dollar system busy counting noses, when it could do this, and so much more, too. Gentlemen, there are giants among us. Let us not stop with killing rabbits, even though that is quite suitable as a way for a hungry man to start.

What conclusions can we draw from these varied experiences, and from the trends discussed earlier? Where do we go from here?

To answer these questions, it is necessary to reexamine the problems we still face.

Management Needs

In general, it seems apparent that the needs of management for information and for communication must be understood, measured, and analyzed much better than they ever have been before. We can hope that the increased interest in management science, in operations research, in mathematical programming, can help us here.

Part of the difficulty arises because accounting data must try to serve in so many ways. It must provide:

1. Data for decision purposes, to aid managers in choosing between alternative courses of action.
2. Data for attention getting, to help managers monitor operations on the exception principle.
3. Data for operating purposes, such as for payroll, billing, credit, cash receipts and disbursements, inventory, and so on.
4. Data for the historical record, such as reports to stockholders, the SEC, tax returns, state licensing, etc.

No program of electronic machine installation can be successful unless it recognizes these needs, and unless the system design integrates the many ramifications of each of these needs. Too often, in my opinion, electronic devices are being installed to gather the same old information, in much the same old way, to process the data in the same old patterns, to produce the same old report, so the same old management can go on filing it in the same old wastebasket—only just a little sooner.

Integration

Integration is not a simple concept.

You have to start down at the grass roots, with the business operations. A good reporting system, in the integrated sense, is not something that views the operations from afar. It is part of the operations. The operations would not be the same without it.

But this is only the beginning. We have to move up a level. Good systems design integrates not only within a function, but between functions. Our objective is to gear the sales to the production, and the production to the sales, and both to the financing, and all of these to the personnel, and so on, throughout the organization.

Still further, for those companies which have plants in different locations, we must have geographical integration. This puts the spotlight on quick, accurate, and not-too-expensive communication.

Still we cannot be satisfied. The whole operation is, or should be, guided and directed by top management, with their methods of planning and control. The systems design must meet their requirements, or there still will be friction and the design can be no more than a partial success.

Finally, and most important of all, we must not lose sight of the social implications. While exulting in the labor-saving that electronics can give, the improvements that it will offer, we must not forget that these changes affect people. Automation of the factory, and of the office, will have such an impact that some are speaking of it as a second industrial revolution.

Whatever part you have to play in this coming change, in the application of electronic computers to business, you cannot evade your responsibility to society. You must consider the people who will use it. You must consider the people who will be affected by it. For example, how can people be retrained so that they can continue to work in this brave new world? What should students do to prepare themselves? How can changes be introduced in an orderly fashion, so they will not upset whole communities?

In the last analysis, the measure of these machines is man. You may design remarkable equipment, and you may install the machines where they can make remarkable cost savings. But unless you have considered your responsibilities, some day your fellow men, your very own children, will look back upon your work and they will not thank you for what you have done.

Automatic Translation of Printed Code to Impulses Acceptable to Computing Equipment

J. T. DAVIDSON AND R. L. FORTUNE†

MR. CHAIRMAN. . . . Ladies and Gentlemen: It is a pleasure for Mr. Fortune and me to be allowed to make this presentation on "Automatic Translation of Printed Code to Impulses Acceptable to Computing Equipment."

It is probable that the best way to cover this subject is to describe a machine system known as the Stanomatic. This name Stanomatic is derived from the words *S*ymbol *T*ranslator *A*utomatically e*N*ergizing *O*ffice *M*achinery. We at The Standard Register Company saw no reason to depart from convention, but we will have to admit that we had a little difficulty finding the words to create a name. It was an effort, but we managed. Accidentally, of course, the name does have some significance when you consider that the Engineering and Research Division of The Standard Register Company developed this device.

Many of you may be wondering why The Standard Register company, with its more than 40 years of manufacturing business forms and feeding devices, has entered into this particular field. A short explanation is in order at this time, without violating the spirit of this conference and its scientific aspect.

We are manufacturers of continuous business forms and feeding devices for application to any type of business machine, such as typewriters, tabulators, etc. Our feeding devices are based on the principle of the pinwheel or sprocket feed. Our forms all carry holes down the sides, which we call Kant-Slip holes, and when these forms are fed into any type of business machine, the holes down the sides of the paper and the pinwheels or sprockets in the machine feed, align, and register these forms automatically and accurately.

We manufacture and carry as stock items some 750 types of pinwheel feed devices, each of them designed to fit the architecture of the machine on which they are installed. We manufacture special auxiliary equipment such as carbon separators, bursters, imprinters, etc. We send out of our plant two to three special devices a day, either adaptations of the standard pinfeed mechanisms or entirely new devices which, up until that time, have not been in existence. We manufacture thousands of autographic registers, the kind of machine you see on the hardware counter, in the jewelry store, florist shops, etc. We do not manufacture tabulators, typewriters, adding machines, bookkeeping machines, or similar types of equipment.

We have for years expounded on the subject of paperwork simplification, and, at times, in trying to carry the gospel of paperwork simplification to business people, we have felt that we were carrying the lone torch in a mass of darkness. Only recently have the businessmen of America begun to realize the intolerable burden of paper handling, and you people making up this conference have attempted to answer this need. Shuffling of papers, the recording and re-recording of information, have become such a burden that (as I believe you all know) there are more people employed in offices today than there are on farms.

We have always liked to feel that we are in a position similar to that of the Process Engineer in the manufacturing plant, except that we confine our efforts to process engineering in the office, attempting to streamline production, so that the end product has been completed as accurately and as quickly as possible with a minimum amount of manual effort. Our nationwide Sales Analysts and Service Departments have made thousands of recommendations for improvement of paper handling, elimination of repeated record making, and, of course, continuous forms and feeding devices for all types of business machines have eliminated the paper handling problem at the machine. Leaving the development, manufacture, and sale of computers and other office equipment in the very capable hands of those persons who are now working on such developments, The Standard Register Company will continue in the field of being the process engineers to American business, attempting to provide the auxiliary equipment and supplies for the input and output of these machines where conventional documents are still going to be necessary.

To those of us who have been associated with office procedures for years and to those who have only recently entered the field, it is a self-evident fact that one of the major problems involved in machine bookkeeping, whether it be mechanical or electrical, is the problem of getting the recording of the original transaction, whether it be retail or another type of transaction, into such shape or form that machines can recognize the recording and use it for the purpose of computation and accounting. The time-honored method of a human being using the eyes and brain to recognize printed material and transferring this information by means of fingers to keys in bookkeeping machines, typewriters, or key punches, will continue to be used, particularly where the volume is not too large. If the preparation of the original document of a transaction automatically created an auxiliary recording in addition to the conventional

† The Standard Register Co., Dayton, Ohio.

recording of the transaction, which could later be used to eliminate the human being translating the conventional writing on the document to keys on a keyboard, and the document could be used directly by a machine to create an accounting record, I believe we would have contributed considerably to automation in the office.

All of us here are familiar with the attempts to produce the ideal, such as the recognition of conventional characters and their translation to electrical impulses, and many others. We do not wish in any way to discredit the very excellent work which has been done in this field. However, Stanomatic, with its simple coding, can be used now in the everyday business world. The Stanomatic (Fig. 1) was developed to accept, and is

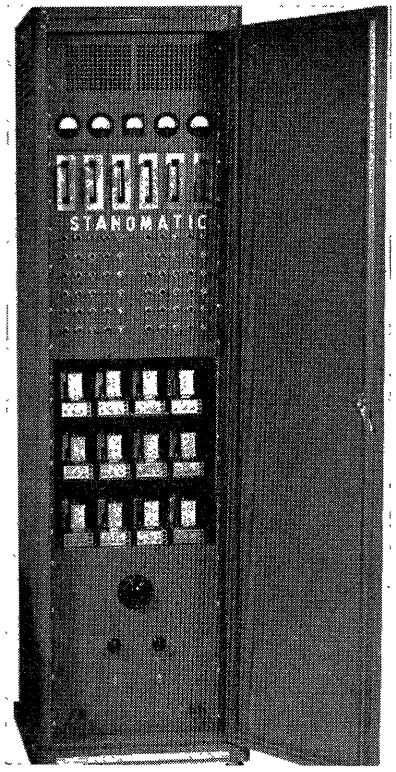


Fig. 1—Stanomatic control cabinet with sensing circuits, control circuits and memory.

capable of automatically accepting, coded information from handwritten or machine-written source documents and translating this information into recognizable business machine code to operate punched card equipment, computer input equipment, etc. From the time the source document is created until the final report is made, there is no need for manual translation from the written word.

In order to better understand the application of Stanomatic to any given system, a greater understanding of the potential of this device will prove helpful. To this end a description of the unit and the basic principles involved is essential.

The Stanomatic senses and translates printed coded information from business forms at a speed of 500 forms per minute. The output of the Stanomatic can be fed either directly into high-speed data-handling equipment or through memory circuits to conventional or slow-speed handling machines. The Stanomatic operates from coded information preprinted with special ink, ribbon, or encoded from special carbon capable of upsetting a sensitive balanced electronic circuit. This coded information, in the form of dots printed on source documents (Fig. 2), is arranged in 30 columns of 5 dot

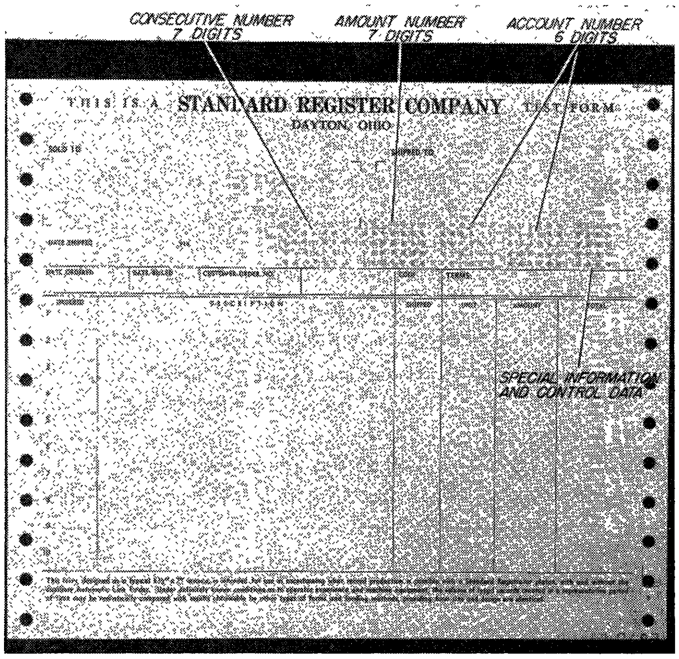

CONSECUTIVE NUMBER 7 DIGITS
AMOUNT NUMBER 7 DIGITS
ACCOUNT NUMBER 6 DIGITS
THIS IS A STANDARD REGISTER COMPANY TEST FORM
DAYTON, OHIO
SPECIAL INFORMATION AND CONTROL DATA

Fig. 2—Form with imprinted code. The code dots may be arranged in any grouping on either the front or back of the form.

locations per column, consisting of 150 bits of information. A single digit in this code occupies one column and its value is determined by the location of two dots printed on a given column. The coded dots release parallel pulses through a predetermined sequence of operations and a signal is fed into data-handling machines as decimal digits. The output can be in any numbering system, binary or similar codes used in electronic computers, 5-, 6-, 7-, and 9-channel punched tape code, or any machine code, by the use of suitable conversion units.

Coding

Stanomatic makes use of 150 bits (printed dots) equivalent to 30 decimal digits per form, which may be used in any combination to provide decimal digits and separate control groups. Each dot represents essentially a binary digit or bit, and the numbers 0 through 9 are shown in the chart (Fig. 3) below with their associated codes. Each printed dot is approximately 1/16 inch in diameter and is separated from any adjacent dot by 1/16 inch. Dots in two of five dot locations in a vertical column are used to represent each decimal digit. The first dot in any column is assigned the value 1; the second, 2; the third, 4; the fourth, 7; and the last provides a self-checking feature. Each digit is composed of two dots so that the translation of their numerical designation is made in self-checking circuits.

DIGIT NO	1	2	4	7	X
	○	○	○	○	○
1	●	○	○	○	●
2	○	●	○	○	●
3	●	●	○	○	○
4	○	○	●	○	●
5	●	○	●	○	○
6	○	●	●	○	○
7	○	○	○	●	●
8	●	○	○	●	○
9	○	●	○	●	○
0	○	○	●	●	○

Fig. 3—Stanomatic code. This is a self-checking code utilizing two code dots for each digit.

The Stanomatic Sensor

At any time after the coding is completed on the forms, they may be inserted into the Stanomatic Sensor. The forms may be fed through the feeder in a continuous strip or as individually cut forms. Fig. 4 shows one

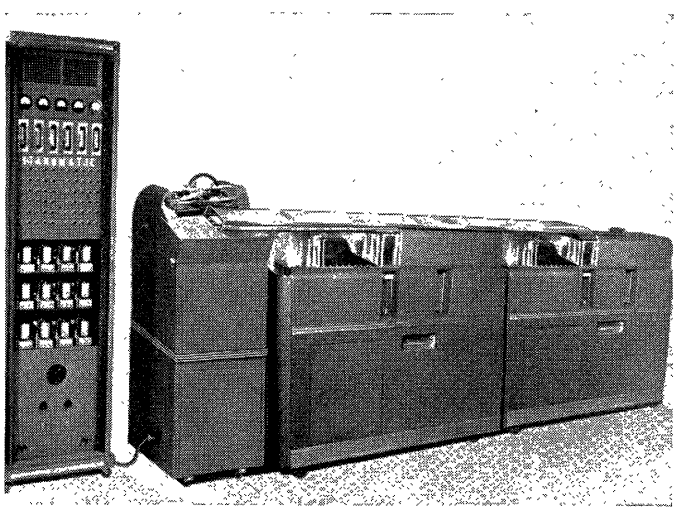


Fig. 4—Stanomatic with two reproducing punches. Operating at maximum speed, the Stanomatic has the capacity to operate five reproducing punches.

variation of Stanomatic equipment where the output from the Stanomatic is used to operate two reproducing punches for the creation of punched card records. The principle of sensing or reading the coded information is the same whether the forms are fed in a continuous strip or individually. As the form is fed, the coded area passes over a sensing head.

The presence of a Stanomatic spot in any given area triggers a balanced electronic circuit (Fig. 5) and the impulses thus created by the sensing units are used to actuate the buffer storage unit in accordance with the code on the source document. Internally, all information is fed in parallel throughout the equipment. At the instant the information is received in the buffer storage, a control pulse is fed through a circuit in the buffer storage unit. This circuit is made by the combined action of the entire code.

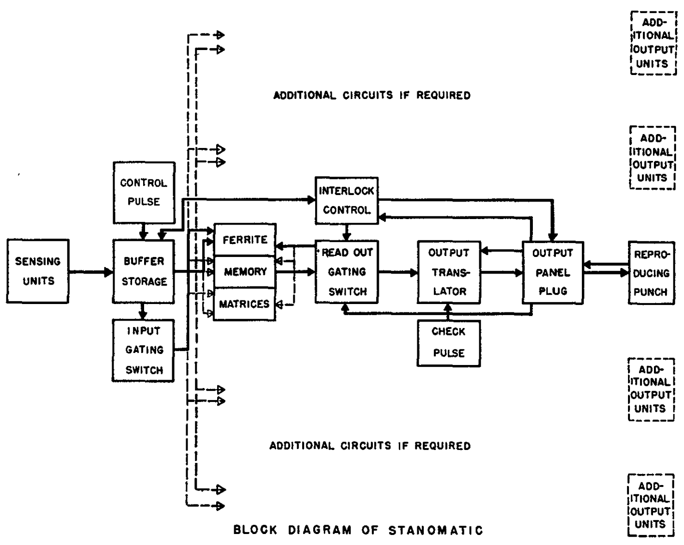


Fig. 5—Schematic diagram of Stanomatic circuit.

In the case where all coded information has been properly read and stored in the buffer memory, the control pulse stores the sensed information into a ferrite memory matrix. The same pulse operates the input gating switch to shift the input circuit to the memory matrix at the next memory plane. This same pulse also triggers an interlock control circuit to actuate a reproducing punch. The final action of the control pulse restores the buffer storage for the next sensing operation. The control pulse, to explain further, is coincident with a pulse of equal magnitude synchronously timed with the operation of the feeding equipment, and storage occurs in the normal manner for ferrite memories. The form just sensed would then be transported to a receiving hopper. In the case of a form being sensed with incomplete information, the control pulse is blocked so that it does not trigger the memory matrix, the input gating switch, and the interlock controls, but triggers and restores the buffer storage unit for the next sensing operation and actuates a reject gate causing the form to be placed in a reject hopper. This feature assures that all information sensed is correct, and that this information will be accurate when received by the terminal equipment, such as the reproducing punch. Rejected forms can be processed by hand, and experience shows the percentage of rejects to be very low.

Associated interlocking controls to properly time and synchronize the operation of the reproducing punch are set up at the time the information is verified by the control pulse. These interlocking controls cause the information stored in the memory to be made available when the read-out gating switch is activated. This switch, in turn, is controlled by the reproducing punch. At the start of operation, the reproducing punch (ready to operate) initiates a read-out pulse immediately after the read-in pulse to the ferrite plane. This information is placed in the output translator and the translator circuits make the information acceptable in decimal form to be punched into a standard tab card. When the first card-cycle of the reproducing punch is completed, the information stored in the translator is erased, the output gating switch reads information from the next into the translator, and the cycle repeats. As information is placed in the translator a control pulse insures that all information originally sensed is still present. If the information is not complete, both the feeder and the reproducing punch stop, indicating trouble. The interlocking system controls a clutch on the source document feeder when all ferrite planes are filled for a given reproducing punch. The feeder will skip feed cycles until an open memory is made available. Necessary circuits to stop both machines in case of jams, misfeeding, etc., are incorporated in the interlocking controls.

Document Originating Machine

Creation of the source documents in the Stanomatic code is accomplished as an automatic by-product of all operations performed on that document from the time it is printed until the document has been completed by the addition of either handwritten or machine-written data. The printing facilities of The Standard Register Company are available for including code imprinting in the Stanomatic ink for serial numbers and other permanent figures at the time forms are printed. If the originating document is to be handwritten, a model of an autographic register (Fig. 6) is available for handling

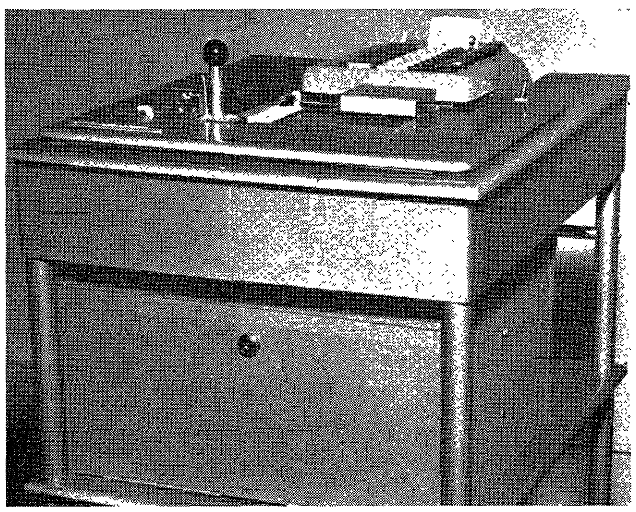


Fig. 6—Imprinting Register. Code impressions are obtained from permanent imprint unit (left) and from keyboard at the top.

marginally punched continuous forms, and for imprinting, in code such information as will be needed to complete the business transaction. This information may consist of the account number of the customer, the branch from which the order originated, the cash amount of the merchandise order, and the specific items required by this customer. Data may be encoded by depressing keys on a keyboard similar to those used on an adding machine. Constant data, such as a branch office, area, or register number, is imprinted from a code slug. In addition, personal account numbers can be obtained by code embossed on credit cards carried by the customer. In instances where there are many points of origin of source documents, a portable unit, Fig. 7, incorporating simplified coding features is available.

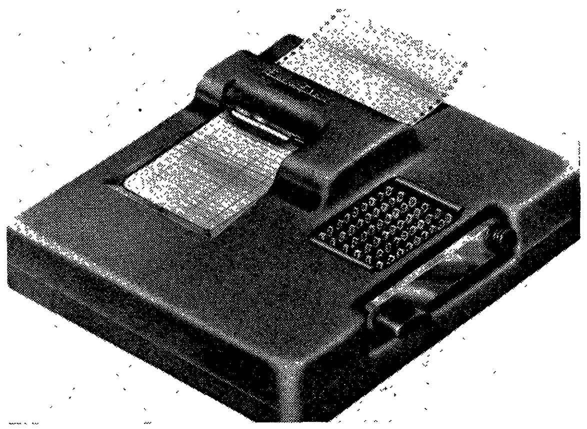


Fig. 7—Portable imprinting register. Code is indicated by depressing keys and the imprint is obtained by depression of the lever.

Two basic units are also available for imprinting codes reflecting machine-written information, depending on the type of form used. On a typewriter or a bookkeeping machine feeding marginally punched continuous forms, the code-imprinting heads mount in a unit directly behind the machine, as shown in Fig. 8, and the

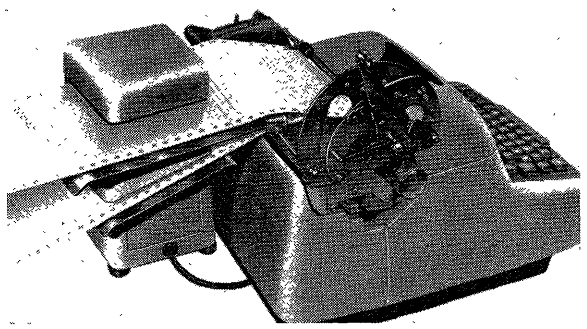


Fig. 8—Imprint unit for continuous form. Imprint is obtained from unit in back of typewriter or bookkeeper, actuated by depression of Line Finder lever on carriage.

individual code wheels index from an electromechanical transfer medium connecting the numerical keys on the typewriter or bookkeeping machine to the code-wheel mechanism. The actual imprinting of the code occurs after a form is ejected from the typewriter or bookkeeping machine. As a result, preselected information to be encoded stores in a buffer memory until a form is ejected. The form is ejected by means of an Automatic Line Finder, which also signals the code imprint unit to print from its memory onto the form. Proper registration in this instance is assured by the pinfeed of the continuous forms.

When individual forms are processed over a machine, the coding mechanism, while similar in design to that above, is not mounted directly behind the carriage, but consists of a separate unit which rests on the typewriter or bookkeeping machine stand, as shown in Fig. 9. The code data are set up in the manner prescribed above and the imprinting is accomplished by inserting the com-

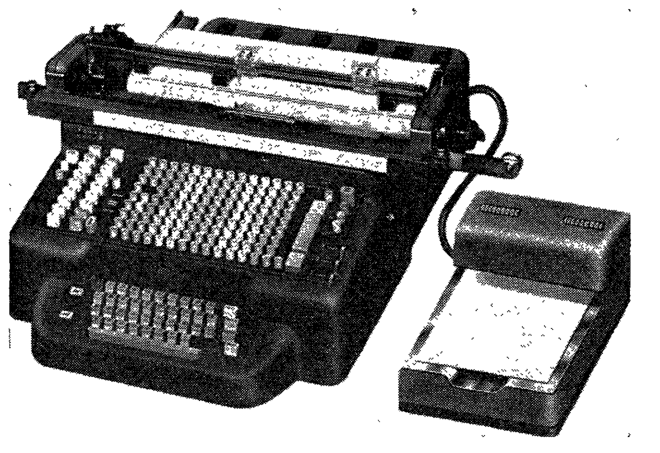


Fig. 9—Code unit for unit zipsets. Carbon interleaved unit zipsets may be inserted in coder to receive impressions.

pleted form into the code unit. The form, when inserted, trips a registration control circuit and the codes are imprinted. Proper registration and alignment are assured by the use of form guides and the code registration circuit. While it is necessary that the code spots be located in relationship to two edges of the source document so that the sensor can interpret data correctly, code can appear at any predetermined position on the form. It need not be in the same order as the typed information, nor must it be located in the same area. If the form receives a great deal of typing, the code could appear on the back of the form, or it could appear in the preprinted heading area of the internal copy, since normal printing ink has no capacity for being sensed. As long as the area imprinted is constant, the coding can appear wherever the customer desires. Code data in excess of 30 digits is encoded on a form in successive groups of 30 digits by advancing the form in the coding machine a suitable distance to give an effect of line-at-a-time printing, and a similar arrangement is provided in the reader and sensor.

Figure 10 brings together the Stanomatic receiving information from a business form encoded by one or more of the input machines mentioned earlier and, in addition, shows the output of Stanomatic flowing to various forms of accounting and computing machines or auxiliary equipment. We have here a small demonstration unit and will move a piece of paper in the sensing zone. This paper has printed on it a dot in sensible ink. When the dot is in position, a light will light, indicating the dot has been sensed. If, in developing the Stanomatic, the auxiliary equipment such as coding registers and other coding means—the inks, carbons, and ribbons—we have contributed to the more ready acceptance of new business equipment and helped to eliminate the laborious and expensive manual transcription of conventional symbols to ones recognized by machines, we will feel that the time and labor have been well expended.

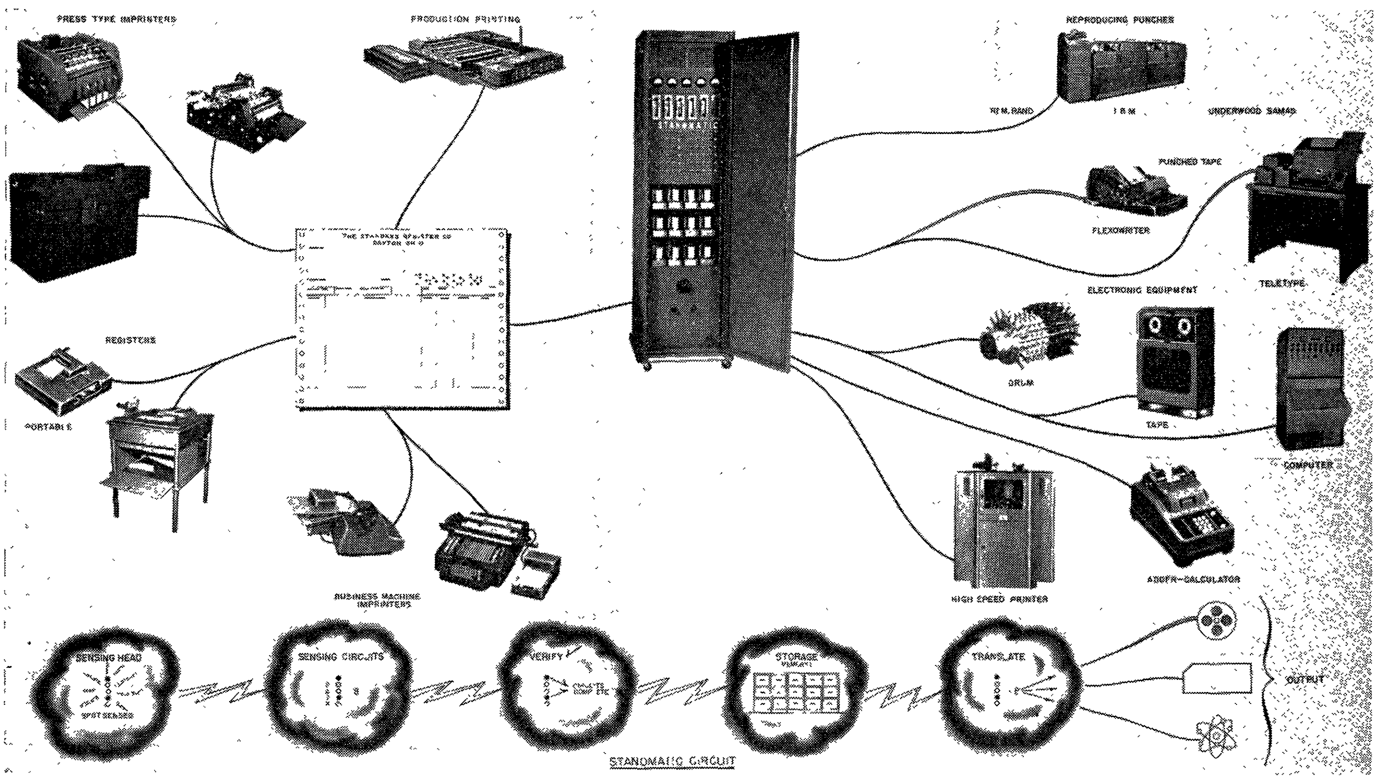


Fig. 10—Stanomatic input, output chart. Code impressions received from the units shown at left may be processed through Stanomatic and interpreted to any code used by the output units shown at the right.

Data Collection as a By-Product of Normal Business Machine Operation

J. C. TAYLOR†

Introduction

THE PURPOSE of this paper is to explain an automatic system of collecting data on a punched paper tape as a by-product of normal business machine operation.

In order to provide a concrete example of such an automatic data collection system, we will describe a system used in a department store to record the data pertaining to the sale of merchandise. Everyone is familiar with the usual role played by the cash register in recording the data pertaining to the sale of merchandise. The price of the article, clerk data, type of sale, etc., are set up on the keyboard and entered into the machine. This gives much information concerning the sale, but with present-day high-volume merchandising, much more information concerning the articles sold is necessary in order to obtain inventory controls, daily sales reports, etc.

First, let us consider for a moment why there is a need for such a data collection system. Before automation it was necessary to perform numerous manual operations in order to prepare various records required by the department store. The necessary reports were prepared by collecting duplicate sales slips, price ticket stubs, cash register receipt stubs, or other media during the day as the sales occurred, and later these media were picked up by clerical personnel who in turn sorted the media by hand and recorded the sales on tally sheets, spread sheets, or punched cards. These records were then processed by hand or through the use of punched card tabulating equipment to produce the reports mentioned earlier. However, during the collection of the information and preparation of these reports, many man-hours were consumed and many opportunities provided for human error and lost sales slips, price tickets or other media. The accuracy of such systems might average 80 to 90 per cent due to the lost media and manual errors. Therefore, the automatic by-product data collection system to be described will illustrate how all these manual methods, with their possibilities of lost media and costly man-hours, will be eliminated by capturing the desired data as a by-product of the normal business machine operation.

One solution to this problem is that of attaching a device to the cash register on the department store selling floor which will capture data pertaining to the sale of merchandise as the transaction is being handled in the normal manner. The cash register used in department stores is a very familiar item; with its classified accumulating features, printed receipt, and indication. Suppose each cash register in our typical department store were equipped with a National Punched Tape Recorder, which will not only capture data describing the sale but also capture data identifying every item sold through each cash register. Such a system will provide all the sales data and identification data necessary to produce the desired reports earlier mentioned. However, some additional information would be desired in order to make the inventory control records more complete. It is obvious that attaching Punched Tape Recorders to other machines used to write buyers' orders, or used in the receiving department for checking merchandise as it is received, will provide additional data which may be processed along with the data recorded at the point of sale and make complete inventory control records. In addition, the tapes from the various machines could also be processed to prepare additional reports and provide statistical data.

As the cash registers or other machines are operated, they prepare their normal hard copy original entry media such as cash register receipts, orders, listing tapes, invoices, etc., in the conventional manner. In no way will the Punched Tape Recorder slow down the operator, although the system is of itself completely sequential in operation, making the system 100 per cent accurate.

Thus one system has been described very briefly by which selected data entered through business machines has been captured as a by-product of the machine's normal operation, while in no way changing the operator's procedure.

The tapes created in this manner may now be applied as input to electronic computers or other data-processing equipment which will prepare the desired reports for the department store management, buyer, etc.

The System in Operation

With the preceding general outline as a background, we will consider in more detail the application of recording the data pertaining to the sale of merchandise. In order to explain the system more clearly, it will be explained in connection with the step-by-step sequential punching of merchandise identification and sales data on a paper tape under control of the cash register. Fig. 1 shows a cash register, recognized as that widely used in department stores, equipped with a Punched Tape Recorder.

The cash register being used in this explanation con-

† The National Cash Register Co., Main and K Streets, Dayton, Ohio.

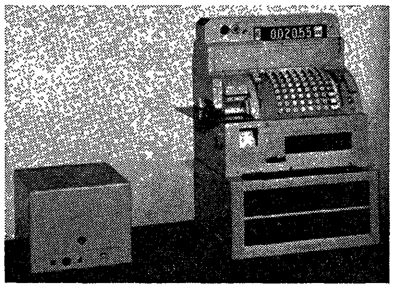
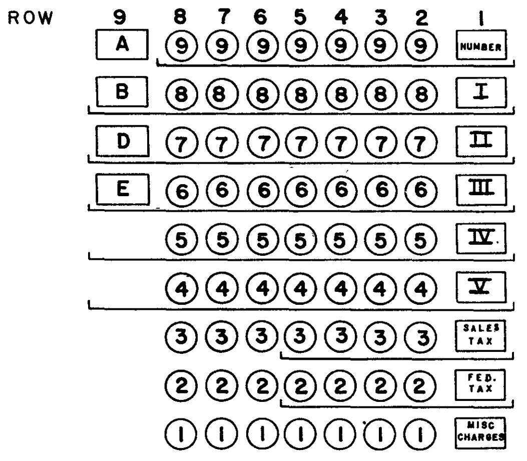


Fig. 1—Cash register equipped with punched tape recorder.

tains a plurality of rows of keys, as shown in Fig. 2, which are used on different operations of the cash register according to the data to be recorded. These data may have different significance in different operations of the cash register. Fig. 2 depicts a cash register keyboard having a total of nine rows of keys. Row 1 is the transaction row, rows 2 through 8 are amount rows which may or may not add into the machine accumulating totals, depending upon the transaction key used, and row 9 is a row of clerk print and totalizer selection keys. The transaction row of keys is used according to the significance of the data being entered on the amount keyboard, and controls the Punched Tape Recorder on each cash register operation. That is, the data set on the keyboard when used with the Number key may represent merchandise-identifying data, whereas when these same rows of keys are used to set data with a Department key in the transaction row the data may represent the amount of the sale or other sales data.



RECORDING SEQUENCE

TRANSACTION KEY ROW 1	FIXED	KEYBOARD
NUMBER	END OF FRAME	ROW1,8,7,6,5,4,3,2
I, II, III, IV, V	REGISTER # DIGITS 1,2,3	ROW-1,9,8,7,6,5,4,3,2
SALES TAX FED. TAX	END OF FRAME	ROW-1,5,4,3,2
MISC.	NO RECORDING	

Fig. 2—Cash register keyboard and recording sequences.

The keys in the various rows of the cash register control the differential positioning of switches according to the data set on the keyboard. The data switches which are differentially controlled are shown in Fig. 3. This figure shows the back of the cash register equipped with the necessary mechanisms and components to control the Punched Tape Recorder. The data switches are positioned by the indicating mechanism in order that the data set on the keyboard be still available after the cash register completes its operation. The indicator, at the top of the cash register, signifies to the customer what was entered through the keyboard.

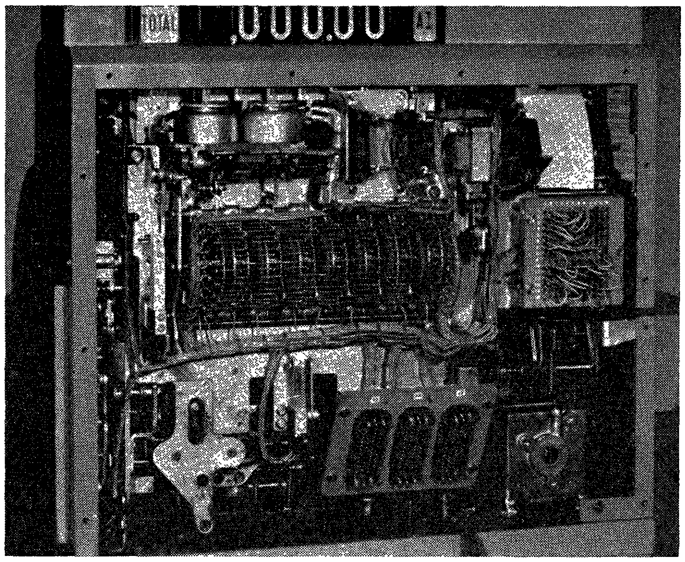


Fig. 3—Back view of cash register.

A means of program control controls the order and number of cash register switches which will enable and control the punching apparatus to produce various punching sequences according to the significance of the data. The keyboard and chart of recording sequences shown on Fig. 2 may be used to explain the program control. Certain representative sequences have been shown for the purpose of explanation. It is possible to program four different sequences for the system, each entirely different, the only limitations being that the word length of each cannot exceed 18 characters, including variable keyboard data, fixed data, and functional codes. The chart of recording sequences illustrates three sequences which are utilized in this particular example.

The brackets on the keyboard chart, which embrace a transaction key and certain other rows of the keyboard, indicate which rows will control the punching apparatus when the particular transaction key is depressed. For example, when the Number key in row 1 is depressed, rows 1 through 8 will control the punching apparatus, whereas when any one of the Department keys I, II, III, IV, or V is depressed, rows 1 through 9 will control the punching. On the other hand, when either of the tax keys, Sales Tax or Federal Tax, is depressed, only rows 1 through 5 will control punching and depression of the Miscellaneous Charge key and will not cause any punching sequence to be followed.

Referring to the recording sequence chart of Fig. 2, it will be noted that when the Number key is depressed the sequence of recording will be first an End of Frame symbol followed by the data set on the keyboard in the following order: rows 1, 8, 7, 6, 5, 4, 3, and 2. If no key is depressed in a row, a zero will automatically be punched into the tape when this row controls the punching apparatus. The End of Frame symbol is a functional code indicating that a new group of data is beginning. All data between two End of Frame symbols will be related to the sale of a particular item of merchandise. This symbol could complete a punched card, return a carriage of a tape-reading typewriter, or control an electronic computer, indicating that a group of information has now been read into the machine and processing may begin.

A second sequence of punching is called for when any one of the five Department keys has been depressed. This second sequence contains a fixed three-digit number (123), representing the cash register number from which the tape was punched. This number is obtained directly from the Punched Tape Recorder each time this program is called for. Next in this sequence is data punched according to the setting of rows 1, 9, 8, 7, 6, 5, 4, 3, and 2, in that order. Therefore, upon alternate use of the Number key and a Department key, a "frame" or group of related data is completed in the form of punched paper tape. The data punched under control of the Number key may describe the item being sold, including the class and style of the article, for example. The data punched under control of one of the Department keys will pertain to details of the sale and will identify the cash register at which the article was sold, what department it was sold in (one of five Department keys), identify the clerk making the sale, and the price at which the article sold.

The sequence of punching data in an operation when one of the tax keys is operated is in the order of an End of Frame symbol followed by rows 1, 5, 4, 3, and 2. The data set up on the keyboard with one of the tax keys is set apart from other data by the End of Frame symbol, as this application does not require that these data be associated with descriptive data, clerk, department, price, etc.

It may be noted that all five Department keys select the same sequence, yet each department transaction is identified by a different digit which is punched under control of row 1 to represent the Department key depressed. In like manner, the two tax keys select a second sequence and are identified by a digit punched under control of the row-1 differentially set switches.

These punching sequences serve only to illustrate the possibilities of punching sequences. One sequence (Number) punches a functional code and variable data controlled from data set on the keyboard. A second sequence controls punching of fixed numerical data and variable data from the keyboard. The third sequence punches only variable data (fewer rows than the first sequence) and a functional code, while another transaction key does not call for any punching sequence.

The program control means, therefore, is controlled from the cash register according to the significance of the data as determined by the particular key which is depressed in the transaction row, row 1, and will control the punching apparatus from only the required number of the differentially positioned switches in the cash register in the proper sequence.

The various components of the program control means and related controls are contained in the Punched Tape Recorder, which is separated from the cash register by a multiconductor cable. Fig. 4 is a view of the interior of the Punched Tape Recorder showing the supply of paper tape which may be seen at the left, the tape then passing across the punching unit, through the punching station, and to the right side of the recorder. Fig. 5 shows the punched paper tape passing from the punching station and being wound on a take-up reel. To the

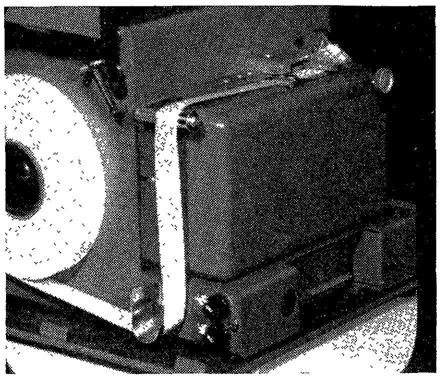


Fig. 4—View of recorder interior (left side).

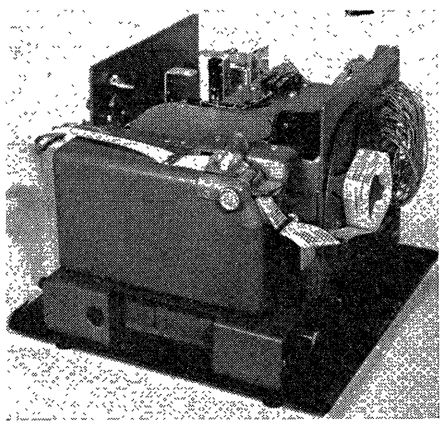


Fig. 5—View of recorder interior (right side).

right rear of the unit can be seen the program board on which the sequences are programmed through the use of jumper wires. To the lower left of the recorder may be seen a "run-in" button which may be depressed to cause tape to be passed through the punching station, providing sufficient length of tape to start on the take-up reel (which also serves as leader to be started through tape reading equipment). As a by-product of advancing tape by means of this button, coded holes may be punched as required by the particular tape-reading equipment which will read the tape being prepared.

The recorder contains control circuits, punch drive motor, paper punch, and a program control means including a scanning switch which is the "heart" of the system.

Inasmuch as when the cash register is operated, all data set on the keyboard are available at the same time, it is necessary to serialize the data when punching the paper tape which will contain data in the form of parallel code and serial digits. The scanning switch operates sequentially, each position selecting one character of data to be punched, and, after punching takes place, moving to the next position to read another digit and so on until the sequence is completed.

The system uses switch encoding techniques, which are a well-known art and therefore will not be discussed in this paper. However, the encoding is done, not directly on the differentially-settable switches but on an auxiliary program board which can be seen to the right back of the cash register in Fig. 3. This board actually represents the termination of the differentially-settable data-encoding switches and is provided for the purpose of encoding by means of jumper wires. One section of the board may be used for encoding the amount and clerk rows while another section may be used to encode the transaction rows. Each of the nine rows of the cash register is equipped with differentially-settable data-encoding switches; the transaction row is equipped with additional switches for the purpose of selecting the proper sequence of punching for the transaction key depressed.

System Block Diagram

The Cash Register–Punched Tape Recorder data collection system is completely sequential as to punching and consecutive entries and therefore is absolutely reliable in its operation. The operation of the system may be reviewed in connection with the block diagram of the system shown in Fig. 6.

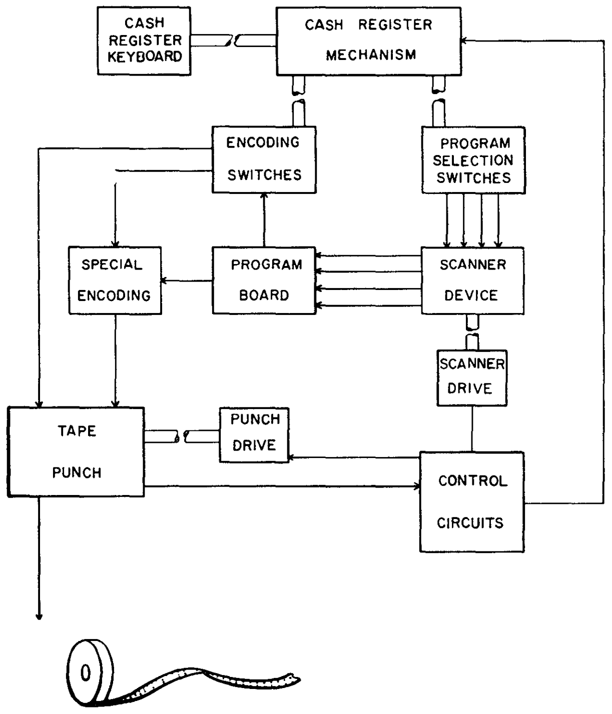


Fig. 6—Block diagram of Cash Register, punched Tape Recorder data collection system.

Setting of data on the cash register keyboard initiates operation of the cash register mechanism. The cash register mechanism, in addition to its normal functions of adding into machine totals, positioning the indication, and printing on the familiar receipt and sales journal, controls numerous components of the system. Among these components are two groups of switches differentially settable and controlled by data set on the keyboard. After these switches are properly positioned, a signal is applied to the group of switches used for the purpose of selecting a recording sequence and known as "program selection switches." If the transaction key used in row 1 on this cash register operation has been programmed on the program board to select a program sequence, that sequence is begun by bringing one section of the scanning switch into use. At this time, if a sequence is selected, the cash register mechanism is locked up under control of the control circuits. This lockup prevents the cash register from cycling again until the sequence of punching is completed. However, the keyboard is left free for setting up data of the next operation.

The program selection switches, having called for a program sequence to be followed, now cause the scanning switch, step-by-step in a sequential manner, to follow through the sequence programmed on the program board. The first position of the sequence applies potential through the program board to either the encoding switches, which have been differentially set by data entered on the keyboard, or to the special encoding network, which provides encoding for all fixed data and functional codes, depending on the programming of the sequence. Through either the encoding switches or the special encoding network, potential is applied to the appropriate electromagnets in the tape punch.

The tape punch being used is a motor-driven unit cycled under control of a single-revolution clutch. As electromagnets are energized, the single-revolution clutch is tripped. The electromagnets, having been energized, also set up conditions to punch the appropriate tape channels as the punch cam line revolves. The punching of the tape feed sprocket holes is automatic with each cycle of the tape punch. As the punch unit cycles and punching takes place, a cam switch applies a signal to the control circuits which in turn controls the scanning switch drive mechanism, advancing the scanning switch to the second position of the sequence involved. At this position the same pattern is repeated, with potential applied either to the encoding switches or special encoding circuits through programming to the tape punch electromagnets, causing coded data to be punched and once again signalling the control circuits to advance the scanning switch to the next position of the active sequence.

When the last position of a sequence programmed to control the punching apparatus controls the tape punch, the signal from the punch once again causes the scanner switch, through the control circuits, to advance one position. This next position (that following the last punching position) controls the control circuit in such a manner as to return the scanner switch to home position preparatory to the next recording sequence, and unlocks the cash register to allow the next cash register operation to take place. Although the cash register has been locked up during a recording sequence, it must again be emphasized that the keyboard has not been locked against operation. The operator may set up data on the keyboard ready for the next operation and the next operation will begin immediately when a transaction key is depressed and the register mechanism is unlocked.

The punching sequences may be selected in any order, repeating the same sequence if desired. Programming of the system is entirely dependent upon the application and the tape-reading equipment. Although there are but four sequences available, this is in no way a limitation of any magnitude. As was seen when the keyboard and recording sequence chart of Fig. 2 were examined, the same sequence may be selected by several transaction keys but differ in significance of data by a digit which identifies the transaction key used. It may also be pointed out that if all four sequences are used and still another is needed it may be possible to make two similar sequences identical. For example, suppose one sequence controls punching in the order of rows 1, 9, 8, 7, 6, 5, 4, 3, and 2 while a second sequence is controlled by data set on rows 1, 6, 5, 4, 3, and 2. It is obvious that these two sequences can be made identical, freeing one sequence, by controlling the second sequence as was done in the first. This of course will punch the tape with data set on rows 9, 8, and 7 which is insignificant for the second type of transaction.

With such a system as just described, a department store sale will be processed in the following manner. The salesperson will first set up merchandise-identifying data by indexing her key in row 9 of our typical cash register, indexing the class of the article in rows 8 and 7, and the style in rows 6, 5, 4, 3, and 2 and depressing the Number key. Depression of the Number key will cause the punching apparatus to be controlled as shown in Fig. 2, punching the class and style into the paper tape. The use of the Number key causes the keyboard entry to non-add. The sales person next indexes the quantity of items in rows 8 and 7, and the gross price in rows 6, 5, 4, 3, and 2, and depresses the proper department key in the transaction row. Thus the second sequence is selected, punching the register number, department, clerk (the clerk key stays down from the previous operation), quantity, and price in the tape. If several different articles are to be sold, each article is handled in a like manner and appropriate taxes and miscellaneous charges are recorded and the machine totalized. The total operation issues the printed receipt for the customer. These operations in handling a sale eliminate the need for accumulating and collecting sales slips, price ticket stubs, or cash register receipt stubs, and also eliminate many manual operations which would normally be necessary later since the tapes may now be fed into computers or tape-to-card converting equipment.

Further Step in Automation

It has no doubt been noted that in the foregoing example it was necessary to enter descriptive data through the keyboard for each item sold. This descriptive data may be of a very few digits, as in our example, and the extra keyboard operation for entering the descriptive data may not be objectionable. However, in high volume merchandising many items may require as many as 25 or 30 digits in order to identify completely the item being sold. Descriptive data of this magnitude might include data identifying class, style, manufacturer, size, fabric, season, color, base price, etc. Since the cash register has a limited number of rows of keys available to enter this data, several operations using the Number key to select a punching sequence would be necessary under the preceding system. The additional keyboard entries necessary in order to record data of this magnitude would be prohibitive in most cases because they would occupy the clerk's time as well as the cash register.

Accordingly, a further step was taken in automation to enable these identifying data to be punched automatically while the clerk operates the cash register in the usual manner to record the data pertaining to the sale of merchandise. This was accomplished by adding a third unit to the Cash Register—Punched Tape Recorder system.

This third unit, known as the National Media Reader, is a device which will read and transfer data from a prepunched price ticket or other media to the paper tape. It is shown in Fig. 7. The reading station of the Media

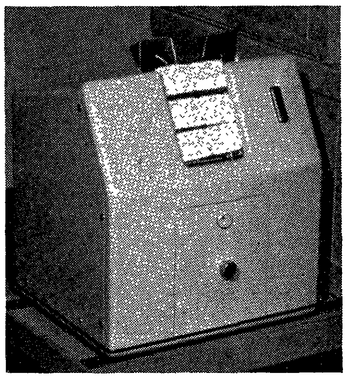


Fig. 7—Media reader.

Reader is seen at the top center of the reader. Fig. 8 shows the three pieces of equipment: Cash Register, Punched Tape Recorder, and Media Reader, all connected by flexible multiconductor cables which allow the equipment to be placed as required by the installation.

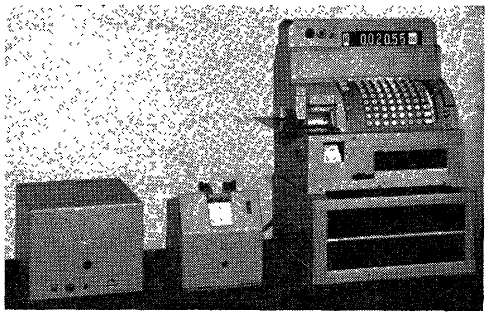


Fig. 8—Cash register equipped with punched tape recorder and media reader.

Fig. 9 shows a prepunched and printed price ticket stub which is being used for this application. These price tickets are usually punched and printed, when articles are ticketed in the marking room, with the descriptive data previously outlined in addition to the original price.

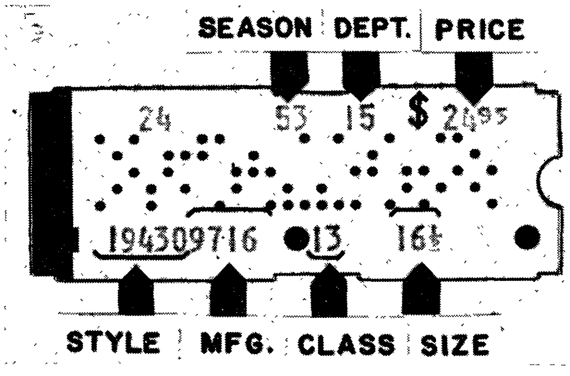


Fig. 9—Typical punched price ticket stub.

The use of the Media Reader simplifies the manner in which a salesperson handles a transaction. Upon selection of merchandise by the customer, the salesperson merely removes a stub of the price ticket and inserts it into the reading station of the Media Reader, which automatically reads the ticket and transfers the merchandise-identifying data to the punched paper tape in the recorder. Along with the reading of the price ticket, certain fixed functional codes such as End of Frame may be punched into the paper tape. While the price ticket is being read, the clerk indexes her clerk key in row 9, the quantity of identical items sold and described by the price ticket in rows 8 and 7, the gross amount of the items in rows 6, 5, 4, 3, and 2, and depresses the proper Department key in the transaction row. The depression of the Department key conditions the cash register so that it can cycle when reading of the price ticket is complete, causing the department sequence, including punching of the register number (provided by the recorder sequence itself), clerk, quantity, price, and department to punch into the tape as well as perform the normal cash register operations of entering amounts into totalizers, indicating the transaction, and printing on the receipt and sales journal. The handling of multiple-item sales is similar to that previously described: the ticket for each different item is read and each ticket reading is followed by indexing of the quantity, price, and department on the cash register. Taxes and miscellaneous charges are entered following the entry of merchandise information. The totaling operation is then performed and a receipt is issued as the transaction is completed.

Suitable interlocks in the system prevent the entering of sales data with the Department keys unless a price ticket has just been read to enter identifying data. The reverse interlock is also true; that is, a second price ticket may not be read until the cash register has been operated to enter the sales data with one of the Department keys. This system of automatically transferring descriptive data to the paper tape under control of the Media Reader and Cash Register does away with the necessity of entering descriptive data through the cash register keyboard.

The Media Reader consists of a motor-driven sensing mechanism, which mechanically senses with pins each character column of the price ticket. The sensing pins are controlled by a single-revolution clutch in sequence, a character column at a time. The five sensing-pins for each price ticket column sense for punched holes. Those pins, finding a hole, pass through the ticket and in turn operate switches associated with each pin. Those switches which were operated by pins passing through the punched holes apply potential to the tape punch electromagnets. Depending upon the configuration of punched holes in the price ticket column being sensed, appropriate tape channels are punched by the tape punch.

Block Diagram of System Using Media Reader

The operation of the Media Reader may be better explained by referring to Fig. 10, which is a block diagram of the Cash Register–Punched Tape Recorder–Media Reader data collection system. The block diagram for this system is similar to that of Fig. 6 except that two blocks have been added to represent the Media Reader. One block represents the Media Reader drive mechanism while the second represents the reading or sensing mechanism. As a prepunched price ticket is inserted in the reading station, the control circuits' sensing the presence of a media renders the reader motor bar effective. When the motor bar is operated, the drive mechanism causes the sensing pins to operate, sensing the first price ticket column. The sensing pins for the first column sense this column for punched holes and operate switches for those code positions in which holes are found. The switches which are operated represent in coded form the digit punched in this digit position or column of the price ticket. They then complete circuits to the proper electromagnets of the tape punch which trip the punch's single-revolution clutch, causing the coded form of the digit to be punched in the paper tape. The revolution of the punch cam line applies a control signal to the reader drive mechanism through the control circuits, causing a further cycle of the reader drive to render operable the sensing pins for the second column, whereupon the sensing means for this second column are operated. Again those sensing-pins, finding holes punched, operate switches, energize electromagnets, and punch the coded digit. Through the control circuits the reader drive mechanism is cycled to cause the next column to be sensed, and transferring of data again takes place. This operation is repeated, reading digit by digit the data prepunched in the price ticket until all desired data have been transferred to the paper tape. At any point during reading it is possible to program the reader to stop transferring data and reset the reader to its home position ready for the next price ticket reading. This is accomplished through an adjustable control which may be positioned to begin reset operation at any point after reading one or more columns from the price ticket. This "reset" control operates a switch which prevents all sensing pins from sensing for coded holes and allows the reader to return to its home position without causing the punch to punch any more data, although the ticket may contain additional data.

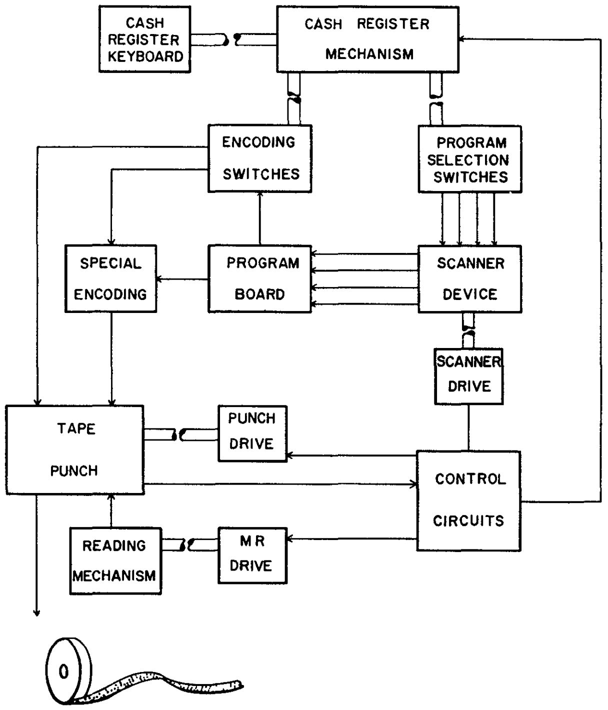


Fig. 10—Block diagram of cash register, punched tape recorder, and media reader data collection system.

Although the reader is returned to its home position, another price ticket cannot be read until a department or price operation is entered through the cash register keyboard. This insures that descriptive data will always be accompanied by the related sales data such as clerk, register number, quantity, price and department information. If the ticket contains a punched price, it might be used only as a base price because the actual selling price, whether marked up or down from the original price, will always be entered on the cash register keyboard. For this reason it is not necessary to repunch price changes in the price ticket.

Although this explanation dealt only with a price ticket application in a department store type application, it is obvious that as the requirement arises, the system can be expanded to include other prepunched media.

There are additional interlocks in the system which add to its accuracy. It has been mentioned earlier that the system is completely sequential in operation. For example, in the event of an open circuit in a data takeout circuit a digit would not be omitted. However, when the sequence calls for punching controlled from the circuit which is open, the punching will stop and the cash register and recorder will remain locked up until the condition is examined and corrective measures are taken.

Another series of interlocks is built around the supply of paper tape. As the supply of paper tape is nearly exhausted, a red warning light is turned on, leaving approximately 15 feet of tape. However, if this warning is ignored the punch will continue until approximately 18 inches of tape is left, at which time the punching stops immediately and the cash register remains locked up. At this time it is necessary to open the Punched Tape Recorder and tear the end of the tape from its pasteboard core. Thus, releasing the tape allows the punching to continue from the point at which it stopped, without loss of any data. The tape left at this point is sufficient to complete a normal transaction. However, if the operator again fails to replenish the supply of tape after completing the transaction, the punch and cash register lock up with $1\frac{1}{2}$ inches of tape remaining, sufficient to complete a normal cash register operation. At this point it is necessary to place a new supply of tape in the recorder. The new roll usually contains about 1,000 feet, which will last for the punching of 120,000 characters.

The paper tape, which is available in 5-, 6-, 7-, or 8-channel widths for various tape codes, is wound on a take-up reel of approximately 400 feet capacity (48,000 characters or about 1,500 transactions). A sample of a five-channel punched tape is shown in Fig. 11. A section of this tape is indicated as representing a price of $12.95.

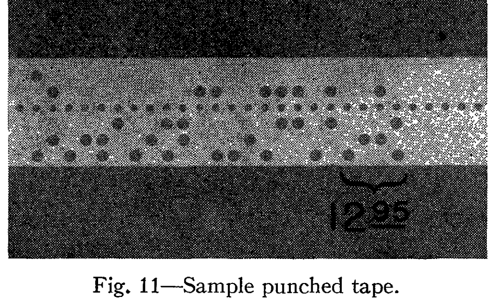


Fig. 11—Sample punched tape.

Although this description has centered about a system using a cash register equipped with automatic by-product data collection equipment, it must be realized that any business machine or mechanism with data settable switches might be equipped with tape recording devices such as have been described. Such machines as accounting machines, adding machines, time clocks, and many others could be equipped with the Punched Tape Recorder. The foregoing description also covered only a very limited application of the system to department store uses. It is obvious that the number of instances in which this automatic by-product data collection system is applicable is almost unlimited.

In conclusion, let us summarize the National Cash Register Company's automatic by-product data collection system. The system captures on punched paper tape, as a direct by-product of the normal business machine's operation, selective data pertaining to the entries through the keyboard of the business machine. As a still further step in automation, a means has been devised by which fixed descriptive data in the form of prepunched media such as price tickets may be transferred automatically to the punched paper tape being prepared by the operation of a business machine equipped with a Punched Tape Recorder. Through the use of this Media Reader it is now possible to automatically transfer data from the prepunched media to the paper tape, while operating the business machine in the normal manner without additional machine operations.

Computers Challenge Engineering Education

F. C. LINDVALL†

WHEN QUANTUM theory was newly stirring the world of physics, an older professor was reported to have said to his graduate students studying this subject, "I don't believe you young fellows understand this stuff any better than I do, but you all stick together and say the same thing." You computer enthusiasts say a great deal, some of which I understand. Your computers differ in detail and complexity, yet have much in common in concept and capabilities and in the fantastic rate with which new models appear. They have their limitations, but nobody underestimates their potentialities. Nor are the glamorous, rosy dreams of coming applications as far beyond the horizon as we might think. You know better than I the magnitude and momentum of the development effort and the tremendous interest which has been generated in all areas of technology and business.

In science and engineering education as well, computers have stimulated much excitement, critical thought, and even some concern. In fact, the computer is one of the more spectacular new developments which are a challenge to education. As a result, we in the colleges of science and engineering are being forced to examine the implications of this challenge. We are also being urged to offer courses in computer fundamentals, logic, design, components, applications, and use, not to speak of complete curricula leading to degrees in computer engineering. Some schools have strong research interests in modern computing and may be justified in offering such instruction. At the same time, other customers for our graduates give equally convincing arguments for more or less specialized instruction in, for example, control systems, instrumentation, automation, systems engineering, operations research, nuclear engineering, and information theory. Needless to say, we are somewhat confused and bothered beyond mere professorial petulance over these challenges to comfortable academic routines.

† California Institute of Technology, Pasadena, Calif.

We cannot, and, I believe, should not, attempt in the colleges to meet these challenges by detailed specialization in all of these new and emerging areas of current interest and importance. We must, instead, do the more difficult job of examining each new development for those features that are truly basic, extracting the concepts that are new and fundamental, and synthesizing the important generalizations that have lasting value. This exercise of self-discipline, sticking to fundamentals, is not easy. The other course, that of following avidly in the classroom the exciting new developments, the intriguing applications, and the fascinating new details, is more fun, has high entertainment value for the student, and is an easy, pleasant way to teach. But, it has the elements of a gold brick, the superficial appeal—the form, but little substance. The values are apt to be transient.

Thus, in appraising new developments, engineering colleges must evaluate critically the fundamental character of these new advances—what is now involved and what is anticipated—so that curricula and course content of the basic sciences and engineering sciences may be improved as fundamental education for future professional application. We must always be critical of that instruction which is specialized training rather than comprehensive education.

To sharpen focus on this problem, we can examine three major areas of computer work which may occur on a college campus: research, computational service, and teaching. These areas, of course, have considerable overlap. Some colleges have already made important contributions in computer research and development. Others have major programs of work, and new activities will undoubtedly develop. These will include: logical design, related mathematics, techniques of physical problem formulation and solution, analysis and synthesis, component research, machine development, and special-purpose computers, to name a few obvious subjects. Much of this activity would closely parallel commercial research and development, but with different objectives that complement the teaching effort and advance fundamental knowledge.

Colleges, through proper research, can contribute more effectively to the fast-moving art of computing than to some of the older, well-established engineering activities.

Computer use on a college campus is relatively new and limited to a few schools, but desired by many. We found at our school that when several departments examined imaginatively the possibilities of high-speed digital computing applied to their existing and potential problems, interest intensified and a computer became a "must." All that stands in our way is two or three hundred kilobucks.

The projected use is as a service center for all campus needs but with individual investigators responsible for their own coding and programming. This general use should develop rather widespread knowledge of modern computation and stimulate investigation of many problems too tough to solve without such computational aid. Studies in computer use will evolve problem formulation, circuit or system synthesis, simulation, special functions, coding, and programming. Data handling and dissemination is another important function—the reduction of experimental data to usable form, thermodynamic properties being a good example.

Formal instruction pertinent to computing can have specific as well as general aspects. Of general character are computer fundamentals including logical design; applied mathematics that incorporates a new philosophy and approach to problem formulation for computer solution; related mathematics, such as Boolean algebra, probability and statistics, group theory, matrix algebra; and mathematics "laboratory" for numerical methods, iteration procedures, and relaxation methods.

More specific instruction could include computer circuitry, components, storage devices, input and output systems, programming and coding, laboratory experience, checking routines, and trouble shooting. However, from the educational point of view, such instruction should be watched critically lest it be too specific to particular computers, devices, and those systems subject to a high rate of technological obsolescence. The significant generalizations can easily be obscured in such instruction by a horde of details which may be only of transient value in the hustling computer art.

Thus we come again to the perennial question which applies as well to all fields of technology as to computers: what can the colleges do best for the student and what can industry do best? We see with increasing clarity that colleges can only begin the education of the engineer—that instruction must be fundamental, broad in scope, and applicable to large areas of technical development. Time should not be diverted to instruction in details that apply only to a narrow segment of industry. Industry itself can supply specific information better than colleges can, and recommends strongly that the colleges concentrate on the basic sciences, the engineering sciences, and the humanities.

The computer's challenge to engineering education is properly in the area of fundamentals. Fortunately, some of the significant basic factors implicit in modern computation underlie other developments in engineering science as well. Probability and statistics, general concepts of reliability, generalizations from information theory, for example, permeate modern engineering, particularly instrumentation and control of all types. Systems study, as contrasted with component study, is the coming approach to involved engineering problems. Computers or simulators of varying degrees of sophistication are likely to be parts of such systems. The science of decision-making may eventually become a recognized engineering science taught in the colleges, which the power of high-speed computing, analysis and synthesis implements. Engineering cybernetics, the more generalized aspects of servos and control, also leans heavily on concepts basic to modern computation.

The implications of high-speed computing in our classical engineering science subjects are a bit frightening. For example, structural design, whether applied to bridges and buildings or to aircraft, could be completely different in approach, if high-speed computing were generally available. Rapid iterative methods, optimization procedures, and even incorporation of nonlinear properties would give an engineer freedom to explore new design concepts. Engineering economy could be greatly extended in scope by computers, in that alternative choices could be examined quickly and tested for the effect of many parameters. In short, the whole approach to a physical problem may be different if high-speed computing is available. The problem formulation would be directly in machine terms from the basic physics without necessarily going through a mathematical formulation. We would then be less inclined to warp the physical system into formal mathematics which we can solve. This would be particularly true if nonlinearities are involved—and most of nature is nonlinear!

Students will be advised of these new horizons in basic thinking which modern computing suggests. However, the great new day is only dawning and we will fumble along for quite a while with analyses and techniques that later may be displaced. Existing methods of engineering will apply indefinitely to thousands of unglamorous but essential problems, and these the students must be prepared to solve realistically. Hence our education in engineering won't suddenly be coded and programmed for the computer art, but will include the fundamental subjects that are pertinent and will develop those broader methods of analyses and thinking that computers will make possible. We welcome the computers with their challenges, but we are not quite ready to throw away our slide rules!

An Optimization Concept for Business Data-Processing Equipment*

D. R. SWANSON†

Summary—**It is demonstrated that under certain conditions there exists an optimal relationship, based on cost and performance, between the number of magnetic tape units and the number of digital computers in a business data-processing system. The relationship is derived in such a form as to indicate how the concept might be generally applied to any kind of system. Some discussion is also given on the question of random access in business data processing.**

Introduction

I SHOULD LIKE to explore some questions which might be of interest to a prospective user of electronic data-processing equipment who is trying to determine the number of magnetic tape units that ought to be included in the system. In Section I, with the aid of well-known mathematical methods, an optimization concept for data-processing systems is developed and generalized, in Section II an example is given in order to show more clearly how to use the results and ideas that have been presented, and in Section III a similar question on the required number of tape units is discussed for a large-scale file maintenance data-processing problem.

I. Development of Optimization Concept

Presently available large-scale electronic business data-processing systems consist essentially of one or more general-purpose digital computers (or "processors"), a number of magnetic tapes, peripheral equipment such as input-output devices, and perhaps auxiliary special-purpose equipment. Reels of magnetic tape are generally mounted on "tape units" by means of which winding, rewinding, reading, and recording operations are effected. Several standard lengths of tape on reels are provided; that most commonly used now is about 2,500 feet.

For our purposes let us suppose that each tape unit can be associated with any arbitrary length of tape. I do not necessarily imply at this point that a tape unit is assumed to be technologically capable of handling, for example, a 10,000 or 20,000 foot reel of tape, but if it cannot do so then it must be assumed that either manually or automatically it is possible to install successively on that tape unit any number of shorter length reels. The total data-storage requirements of the business operations to be mechanized determine, of course, the total length of magnetic tape that must be in the system. With the foregoing assumption, however, the number of tape units associated with the given total length of tape is not fixed but can be selected on the basis of the cost and performance considerations to be discussed. A noteworthy point, incidentally, is that nearly all of the cost of a magnetic tape memory lies in the tape units rather than in the tape itself.

Let me first try, by considering a specific example, to make intuitively clear the optimization concept to be formulated. Suppose the total amount of data to be stored requires 100,000 feet of tape. On the one hand, we might provide a system with ten tape units, each handling 10,000 feet of tape or, on the other hand, we might provide 1,000 units, each containing a 100-foot reel of tape. It is evident that the latter alternative would be a great deal more costly than the former, but would have the advantage of much lower access time to information recorded on the tapes. The question of whether or not the increased performance of the latter system is worth the extra cost is extremely difficult to answer because it apparently requires knowing the dollar value of faster data processing. To see how this question is avoided, let us pursue the example somewhat further. Suppose that we add to the first (ten-tape-unit) system enough (perhaps ten or twenty) computers or processors so that its total cost is brought up to that of the second (1,000-tape-unit) system. For the same price, then, which system can handle and process data more rapidly? Or can a much better system be put together for that price, consisting perhaps of several processors and a few hundred tape units? The best system is certainly that in which a balance exists between tape access time and the effective computing speed of the several processors, so that neither tapes nor processors can be considered a bottleneck. To determine such an optimal balance requires some knowledge of the nature of the data-processing jobs to be done, but it is not necessary to know the dollar value to the user of mechanizing those jobs.

With the foregoing qualitative introduction, the problem can now be described in abstract mathematical terms, not entirely for the purpose of being obscure but rather to divorce the optimization concept from a par-

* Most of this paper was prepared while the author was associated with the Hughes Research and Development Laboratories, Culver City, Calif.

† Ramo-Wooldridge Corp., Los Angeles, Calif.

ticular kind of system in order that its more general implications will not be overlooked. In the following paragraph, in fact, the word "system" is used in a very broad sense; it is not necessarily restricted to hardware and may refer, for example, to groups of humans and machines in a business or economic environment.

Consider, then, a system for which performance depends upon the degree of complexity or number of components (hence the cost) that one is willing to put into the system. Cost and performance are then complementary in the sense that one may be improved only at the expense of the other. It is possible that a similar analysis using other pairs of complementary quantities, speed and reliability, for example, will be found useful for certain systems. In fact, more than two quantities might be involved. The problem may be stated generally as follows: We have a set of variables $(x, y, z, \cdots)$ which in some way characterize the system, and wish to maximize a function $f(x, y, z \cdots)$ subject to a given set of constraints: $g(x, y, z, \cdots)=0$, $h(x, y \cdots)=0$, etc. If the variables are taken to be continuous, and if certain other conditions are satisfied, then the solution can be obtained using well-known techniques (method of Lagrange multipliers) of the calculus of variations; I shall not write the general solution here since it appears in standard mathematical texts.[1] The particular case of two variables and one constraint yields the following condition for an optimal relationship between x and y:

$$\frac{\partial f}{\partial x}\frac{\partial g}{\partial y} = \frac{\partial f}{\partial y}\frac{\partial g}{\partial x}. \tag{1}$$

From the symmetry in f, g, it is clear that optimizing g, with constraint $f(x, y)=$constant, yields the same result. Whether the stationary point reached is maximal or minimal, and whether or not it is the only one within the allowable ranges of the variables are questions most easily answered when the specific form of the functions f, g are known. If the $(x, y, \cdots)$ are actually discrete variables, a trial and error method in the neighborhood of the solution to (1) can be used to obtain the desired optimum.

It is, incidentally, not assured, even having successfully formed f, g, h, etc., that a solution in the form of a stationary point actually exists. For example, if $f(x, y, \cdots)$ and all constraints are linear, then the problem is one of bounded variation which is not amenable to the infinitesimal processes of calculus, so that linear programming methods must then be used. Still other problems, for example those nonlinear problems in which boundary conditions play a role similar to that in linear programming, may not be solvable at all. No attempt will be made to discuss here the criteria that determine into which, if any, of the aforementioned categories a given problem falls, but I suspect that very often this question can be answered through an intuitive understanding of the physical situation which the problem represents.

[1] R. Courant and D. Hilbert, "Mathematische Physik," Julius Springer, Berlin, Germany, vol. I, p. 140; 1931.

II. Application to General Accounting Model

To return to the example presented earlier, let x = number of processors, y = number of magnetic tape units, $f(x, y)$ = time required per data-processing job, averaged over all jobs to be mechanized by the system; f is therefore actually an inverse measure of performance. $g(x, y)$ = cost of system.

Now, at this point we need detailed information about the equipment in order to write a cost function $g(x, y)$, and about the nature of the data-processing tasks in order to determine the form of $f(x, y)$. To write functions f, g which have some general validity is of limited value since, when applied to any specific case, they will not be as satisfactory as functions tailor-made to fit that case. However, in spite of the limited applicability of the final results, I think it will be instructive to follow through at least one partially realistic example.

First, let us assume that the tape units permanently carry reels of any length tape, and that they are fully automatic; otherwise in $g(x, y)$ would have to be included the cost of hiring a man to load and unload reels and in $f(x, y)$ any time delays while he is so doing. Assume further the cost of the system to be linear in x, y.

$$g(x, y) = Cx + Dy + E, \tag{2}$$

where C = cost of one processor, D = cost of one tape unit, and E = any equipment cost that is independent of the number of tape units or processors. This form of g implies that there are no "quantity discounts," and that the cost of a tape unit is independent of the length of tape which it carries.

For the purpose of determining an $f(x, y)$ that has at least some measure of general significance, and at the risk of appearing to oversimplify an enormously complex question, I have attempted to piece together a description, suitable for our purposes here, of that part of business data processing which arises from general accounting procedures.

The bulk of the data processing from general accounting occurs in posting entries to various files of subsidiary accounts and the subsequent processsing of those files of accounts to produce reports and ledgers of various kinds. We distinguish therefore three fundamental kinds of records: "entries," "accounts," and "reports." Rather than attempt a rigorous definition of these terms, let me simply present an example, then base definitions in a general way upon the specific significance quantities take on in that example.

A daily or weekly timecard is an "entry" to a payroll "account." The active career of the entry document is relatively short; its principal mission in life is to be posted to one or more accounts; it may then be consigned to inactive archives or perhaps destroyed. The account is of a relatively more permanent nature and contains a running record, in summary form, of various pay and deduction transactions of the employee. Since the account must be kept in a file with other similar accounts, corresponding perhaps to thousands of employees, it is characteristically sorted to some sequence on a key word such as name of employee or payroll number. Therefore, if an entry originates at random with respect to that sequence, a sorting operation of those entries accumulated over a period of time must precede the posting. (An example of this would be a storeroom requisition which is posted to an inventory account. The time sequence in which requisitions originate will, of course, bear no relationship, in general, to the part number sequence in which the inventory accounts are filed.) Reports will be defined in a very broad sense as the outputs from a file of accounts. Such outputs may take the form of other subsidiary accounts or ledgers which contain highly digested summary information.

In terms now of how we assume our electronic data-processing system to be used, the foregoing outline of general accounting operations can be described as follows: Files of entries and files of accounts are accumulated on various magnetic tapes within the system. To keep our example as simple as possible, it will be assumed that the operation of sorting does not tie up the processor but is done by separate equipment, perhaps by punched card methods, so that entries are all sorted to the proper sequence before entering the magnetic tape-processor system we are considering. Collating, or merging, posting, and possibly some reporting or other processing operations for a given file of entries and corresponding files of accounts can be done in a single combined run of several tapes, provided, of course, that all such account files are sorted to the same sequence. (With a general-purpose computer as processor, there is probably not much point in performing these operations separately from one another as is often done with punched card tabulating equipment.) Since the total amount of information stored in all of the entry and account files determines the total length of tape required, and since we are solving for the optimum number of tape units, it must be noted here that one tape unit may carry many files of accounts or entries. Accordingly, the actual procedure of bringing into the computer and processing corresponding individual entries and accounts must be preceded by positioning both the tape which holds the entries and the tape or tapes which hold the corresponding accounts, so that the beginnings of the files to be processed are immediately accessible to the computer. Since $f(x, y)$ represents the average length of time required to carry out a posting operation (and perhaps other processing) with a given set of one entry and one or more account files, it is clear that it consists of two terms. One represents the tape running time required to make the files accessible to the computer, and the other represents the time required to effect the operation itself.

Let

A = Average overall file operations of:
Fraction of tape on one unit which must be run through prior to a given posting operation, and while the processor is idle, to make beginning of file accessible to the processor. If several tapes are running simultaneously (one entry file and perhaps two account files), then A corresponds to the unit that reaches its destination last. From the definition, A lies between 0 and 1.

Bt = Average overall file operations of:
Time required to execute all posting, reporting, and miscellaneous processing that is carried out once the appropriate files have been made accessible to the processor in a given operation.

N = Total number of files (entries plus accounts) in system.

t = Time required to run a magnetic tape unit continuously through a length of tape corresponding to an average-length file.

Then, with x = number of processors = 1, and y = number of (equal length) tapes in system,

$$f(1, y) = \frac{ANt}{y} + Bt. \tag{3}$$

It has been assumed that no part of an operation is begun until the preceding operation is entirely finished; otherwise the two terms in (3) may not be strictly additive. The effect of adding more processors to the system depends on just how the processors are used. One possible way, but perhaps not the best, is to share among the several processors the internal processing time on each entry-account pair, with tape access and read-in time unaffected. Therefore, let $B = K + J$, where Kt = time required to transfer entry and account information from tapes to processor, Jt = internal processing time once transfer has been effected. Therefore,

$$f(x, y) = \left(\frac{AN}{y} + K + \frac{J}{x}\right) t. \tag{4}$$

Other (probably small) terms may be present depending on the detailed logic of the program used to allocate work to the several computers. Eqs. (1), (2), and (4) yield

$$\frac{\text{No. tape units}}{\text{No. processors}} = \frac{y}{x} = \sqrt{\frac{ANC}{JD}}\,. \tag{5}$$

C/J may be taken as some kind of "cost-speed" index for the processor, while D/A is an analogous quantity for a tape unit. The greater the ratio of the former to the latter, the greater is the optimum number of tape units to associate with one processor, as might reasonably be expected.

With a processor-to-tape-unit cost ratio of $C/D = 25$, $N = 50$ files of accounts, and $J = 4$, we have $y/x = 18\sqrt{A}$. ($J = 4$ might be obtained as follows: If, for each entry-account pair it is necessary on the average to spend 80 milliseconds of internal computing time, and if the rate of information-handling from tapes is 10 characters per millisecond, so that a 200-character account requires about 20 milliseconds, then internal processing time per file averages about four times the time required to run through an average length file on tape.) Since $0 < A < 1$, the optimal number of tape units per processor in this example is less than 18. It may be possible to arrange information on the tapes and to plan and maintain a schedule of tape usage that almost eliminates necessity for "wasting" time positioning the tape before each run; hence, A may be a small quantity. In that event, the actual number of tapes may be governed by considerations other than those presented here; for example, the desirability of having entry files and all account files corresponding to a given entry file on separate tapes. It is more likely, however, that such rigid plans and schedules cannot be maintained, since programs and even output requirements will undergo frequent change, disruptions will occur because of equipment breakdown, programming errors, failure of inputs to be generated on schedule, etc. If the files were placed randomly on the tapes, and if it were necessary to position only one tape, A would of course have the value 1/2. If two tapes were involved each time, then the average fraction of a tape length that must be run before both are in position (provided they can be run simultaneously) is 2/3. If three or four tape units were used in each operation, A would have a still higher value. With a placement of files intermediate between random and ideal, and assuming several tapes per operation, it is reasonable to put $A = 1/2$, so that $y/x = 13$ tape units per processor. With this result, and using the performance requirements to determine a numerical value for $f(x, y)$, it is possible to solve explicitly for x and y. The value obtained will not necessarily be integral so can be used only as a guide to the desired optimum.

In the example outlined, let us see what the penalty might be for using a nonoptimal system. Suppose that the system in question consists of one \$500,000 processor and fifty tape units which cost \$20,000 each; total cost \$1,500,000. With $K = 1$ as a reasonable value, and again $C/D = 25$, $N = 50$, $J = 4$, and $A = 1/2$, we have

$$f(x, y) = \left(\frac{25}{y} + 1 + \frac{4}{x}\right) t = 5.50t.$$

Now, with this same performance, build a system from two processors and some smaller number, y, of tape units:

$$f(2, y) = \left(\frac{25}{y} + 3\right) t = 5.50t.$$

From which, $y = 10$ tape units; (each unit, of course, holds five times as much tape as in the first system). According to our earlier result ($y = 12.5x$) the system is still far from optimal, but with the same performance the cost is now only \$1,200,000. If fifteen more tape units were added, so that the system is optimal, then, for a cost of \$1,500,000, the performance is $f(2, 25) = 4.00t$. Thus, for the same cost as the (1, 50) system, the (2, 25) system can do the same amount of data processing in 73 per cent of the time.

The foregoing example is useful only to illustrate a method of procedure; the results themselves are, of course, severely limited in applicability because of numerous explicit and implicit assumptions that have been made on the nature of the equipment and the data-processing operations. However, it is generally true for magnetic tape memories that cost increases with decreasing access time; in that circumstance, then, an optimum balance between tapes and processors must exist.

III. Random Access and File Maintenance

Finally, I should like to make a few comments on file maintenance and random access to large magnetic tape files. We shall consider only a single large file of accounts, the processing and maintenance of which essentially keeps busy one computer. This problem is important in the field of business data processing and does not fall within the framework of the multifile accounting model discussed in Section II. Examples are customer accounts receivable for a department store, stock balances, and other records for a large inventory of parts.

Each account within the file is assumed to be uniquely identifiable by a single key word; for example, customer name, customer account number, part number, etc. The file is sorted, either numerically, alphabetically, or both, to the sequence of the key word. Input information, such as transactions to the account or changes of basic information within the account (e.g., change of address) nearly always originates at random with respect to the file sequence. To process that information, therefore, involves three steps: (1) search for the appropriate account, (2) transfer that account to the internal memory of the computer and process the transaction, change, or entry, and (3), record the updated information back onto the magnetic tape files (either in its original position or on a new tape, depending on the capabilities of the equipment) or perhaps transfer the information to a printer output if the input was in the nature of a request for information.

It is first important to recognize that the originating of some kind of transaction for a randomly selected account is not an isolated event, but is one of a more or less perpetually recurring series of such events. Over a period of time, then, input transactions originate for a large number of accounts within the file and the usual procedure of periodic sorting and merging into the main file can be followed. The longer the period of time over which inputs are accumulated the greater the fraction of the file that is affected; therefore, the longer a time one is willing to wait before the file is updated, the more efficient (in terms of computer usage) becomes the file maintenance processing. Thus what begins with random inputs finishes as a sequential process. To put some of these concepts on a quantitative basis, let us first define the two fundamental quantities which describe those properties of our data-processing system that are relevant to the question of random access and file maintenance.

1. "Response time," denoted by t, of the system is the interval between the time a request for information, a change, a transaction, or other kind of entry, enters the data-processing system and the time the system either responds to the request or simply is brought up to date so far as the information in the account affected is concerned.

2. "Traffic capacity," denoted by Q, is the maximum number of input transactions per unit time that the system is capable of processing. The inverse of Q is the time spent per transaction and would be equal to the response time if only a single transaction at a time were in the data-processing system.

These two quantities can be visualized most easily by considering an analogous situation of a one-way tunnel carrying automobile traffic. The "response time" is the total time any one car spends within the tunnel, the "traffic capacity" is the maximum number of cars that can leave the tunnel per hour during a "steady state" in which just as many cars are entering as leaving.

Knowing what the data-processing problem requires in the way of response time and traffic capacity, we should like to know how many tape units to use in the system. Again we assume, as in Part II, that it is technologically possible for a tape unit to carry a reel holding any length of tape up to the maximum amount carried in the system. It is also taken for granted that the cost of the tape memory system will increase with the number of tape units into which the total length of tape required is divided.

The processing operation we are considering can be divided into cycles of uniform duration t. Those information inputs or inquiries accumulated during one cycle are processed during the next. Each cycle has two phases: "sorting" of inputs accumulated in preceding cycle (which requires a time s), and "merging" of the inputs into the file of accounts (which requires a time m). A convenient unit of time is that time required at normal tape-reading speed to run through one average account length on the tape. (For a 200-character account and 10-character-per-millisecond tape reading speed, and allowing for time required to traverse blank space between accounts, this time unit would be about 30 milliseconds.) If there are M accounts in the system, then, in the time units chosen, M also represents the time required to run through consecutively the entire magnetic tape file. If there are n equal-length tape units in the system, the running time for any one unit is M/n; however, let us assume that somehow it is possible to run all tape units simultaneously and to stop them all simultaneously when an account to be processed is reached on any one of the units. The merging time is then given by the following kind of equation: $m=(M/n)+jqt$, where the first term is the time required to run through all account files on n tapes simultaneously, and the second term is the time spent processing those accounts for which entries have accumulated. q is the peak rate, averaged over a time t, at which entries come into the system; qt the number that accumulated in the previous cycle time, and j the average time required to merge and process each account once the account is immediately accessible to the processor. The possibility is ignored that some accounts might have more than one entry accumulate; this amounts to the assumption that $qt \ll M$. Similarly, $s=kqt$ where k is the average time per entry spent in sorting the entries. (For example, if any one decimal sorting pass on a magnetic tape sorter proceeds at tape speed, and if 10 output tapes were available per pass, then k is just the number of digits in the key word, i.e., the number of passes.)

Thus, we now have

$$t = s + m = \frac{M}{n} + (j + k)qt.$$

The maximum value which q can have is clearly $1/j+k$, since t must be a positive quantity; but that value is therefore the traffic capacity, Q, of the system.

$$Q = \frac{1}{j + k} = \text{traffic capacity,}$$

$$t = \frac{M}{n\left(1 - \frac{q}{Q}\right)} = \text{response time,}$$

j and k can be determined only after rather detailed programming of the problem; in any case, it is interesting to note just how the number of tape units required depends on both the traffic capacity and the response time.

$$n(t, Q) = \frac{M}{t\left(1 - \frac{q}{Q}\right)} = \text{minimum number of tape units required.}$$

To understand better how to use those equations to determine n, consider a specific example:

The quantity j is quite similar to the quantity $J+1$ of Section II, so we may take $j=5$ as a not unreasonable value. If the sort is on seven decimal digits, then possibly $k=7$ as suggested earlier, so that $Q=1/12$ of an entry per our unit time, or roughly three entries per second. Suppose that the actual peak entry rate is $q=2$ entries per second, so that $n=(3M/t)$. If the number of accounts M is such that to run through them consecutively on tapes takes 2 hours (a time which corresponds to perhaps $\frac{1}{4}$ million accounts at an average of 200 characters per account), and if the maximum acceptable response time is 20 minutes, then the system should contain $n=(3\times 2)/(1/3)=18$ tape units. At a commonly used recording density of 100 characters per inch, each tape would be on the order of 3,000 feet long.

Having demonstrated a set of conditions and some criteria under which magnetic tape files, fundamentally a sequential type memory, can be used for a so-called random access problem, the question might now be raised as to what kind of business data-processing problems require a large-scale random access (e.g., matrix type as opposed to sequential) memory, or for that matter even a very rapid access memory. To begin with, the several-per-second traffic rate chosen in the example is not greatly exceeded by peak rates of credit transactions of customers in a large department store or even by stock requisitions to Air Force inventory accounts at large depots. These two examples are isolated, of course, but they serve as convenient illustrative reference points. Secondly, business data-processing "response times" are more likely to be days or hours rather than minutes or seconds; probably seldom less than the 20 minutes chosen in the example. It is certainly possible to point to a few business operations where very short response times are useful (these examples generally seem to involve a customer, in person or on the phone, who must not be kept waiting, e.g., credit authorization for department store customers), but even in these cases it is not always necessary to refer to a voluminous file. If it could be economically justified, a large rapid access memory for sorting would of course be quite useful in many business applications. I suspect that the requirements of many other so-called random access problems in business actually can be met with magnetic tape equipment.

Data-Processor Requirements in Production and Inventory Control

H. T. LARSON AND A. VAZSONYI†

Introduction

AUTOMATIC DATA processing, as performed by punch card machines, has been an important tool in the business world for many years. In the field of accounting and statistical data collection, it would probably be impossible to conduct business effectively, without the use of such machines. During the last few years it has been increasingly recognized that new electronic computers, through their superior performance, will surpass in efficiency the current punch card machines. A number of firms are in the process of introducing electronic computers in their business operations, and an even larger number of organizations are making plans for the application of large-scale electronic computers.

Automatic data processing in production and inventory control, on the other hand, is something of a new development. There are a number of firms using punch card machines in production and inventory control, but it appears that there is no corporation that has as yet a fully integrated automatic production and inventory control system. There are a number of reasons why progress has been relatively slow in this field. First of all, problems in production control are a great deal more complicated than, say, problems of conventional accounting. Furthermore, the methods of production and inventory control have been only recently developed, systems and procedures have not sufficiently settled down, and therefore there is a continuous need for the introduction of new techniques. Another point is that the techniques of production control are not yet sufficiently articulated; and, consequently, procedures often are carried through on an intuitive basis. This makes the introduction of automatic data processing exceedingly difficult.

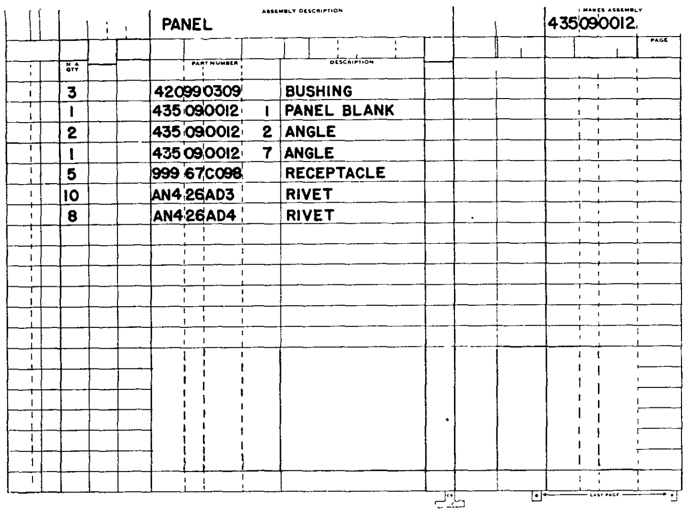
To appreciate the complexities of production and inventory control, compare, for example, the problem of preparing a payroll with the problem of parts listing. True, there is a great deal of data processing in the preparation of a payroll; however, the problem is conceptually quite simple. A clerk in payroll accounting can be trained in a short time; the principal difficulty is the problem of handling the large amount of data accurately and speedily. Methods of avoiding and finding errors in this type of accounting system have been worked out for many years. Compare this situation with the problem of parts listing for a large manufacturing organization. Again, there is a great deal of data, but there are many conceptual complications. Complex assemblies are made up of other assemblies and of simple parts; improvements in engineering design continuously make obsolete some of the parts; inventory levels constantly change and adjustments must be made; sudden changes in master schedules must be made to correspond to new customer requirements. Many other similar features make data processing in production control exceedingly difficult. The development of checking procedures to assure the accuracy of the results obtained in

† Ramo-Wooldridge Corp., Los Angeles, Calif.

itself presents a problem much more complicated than the checking problems usually encountered in more conventional types of data processing.

It is not surprising then that during the last few years there has been an increasing emphasis in this field and it is gratifying that progress has been made at least on two fronts: (1) performance of electronic data processors has been evaluated by various organizations from the point of view of production and inventory control, and (2) new scientific methods, in particular mathematical models, have been developed to describe and analyze the processes of production control. The purpose of this paper is to report on the work of the authors in this field and to show through an illustrative example how some of the basic problems of production control can be handled with the aid of mathematical models and how electronic computers can be applied to these problems on the basis of such mathematical models.

Imagine a hypothetical factory producing some rather complex assemblies, similar to airplanes or radar sets.



ASSEMBLY DESCRIPTION: PANEL

MAKES ASSEMBLY: 435090012

PAGE

N. A. QTY	PART NUMBER		DESCRIPTION
3	420990309		BUSHING
1	435090012	1	PANEL BLANK
2	435090012	2	ANGLE
1	435090012	7	ANGLE
5	99967C098		RECEPTACLE
10	AN426AD3		RIVET
8	AN426AD4		RIVET

LAST PAGE

Fig. 1—Assembly parts list. This sheet refers to the assembly "panel" which is assigned the part number 435090012. This panel is made up of seven different articles which could be subassemblies or parts. The panel requires 3 bushings 420990309, and 1 panel blank 435090012-1, etc. Each assembly will have a similar sheet.

There might be thousands of these articles produced on assembly lines and in the machine shops. Imagine that master schedules for the shippable items are set well in advance, but are subject to periodic changes. The problem we propose to examine here is how to determine the number of parts and assemblies required to meet this shipping schedule and when these various parts should be manufactured. A further problem to be answered is the determination of machine and labor hours imposed on this factory by this master shipping schedule.

We develop answers to these problems through the development of a mathematical model. It will be seen that the mathematics involved are matrix multiplication and inversion. Fortunately, due to a special property of the matrices involved, the problem of inversion will appear as a problem in matrix multiplication. The principal part of the computational procedure being matrix multiplication, we spend a considerable part of our paper on this problem. This might lead to the conclusion that the need is for a special-purpose computer and not for a general-purpose data processor. This conclusion would be in error, as production control involves a great deal of conventional data processing such as sorting and collating. Furthermore, production control usually must be tied in with cost accounting and labor distribution, and it is unlikely that the computer would do parts listing and scheduling exclusively.

The exposition of the paper falls into two natural parts. The first is the development of the mathematical model. The second describes the computational technique involved, including estimates of the magnitude of the job to be performed for typical examples and also certain of the computer characteristics required.

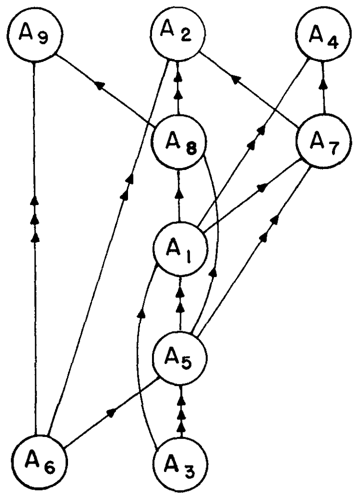


Fig. 2—The Gozinto graph is a pictorial representation of the parts requirements. The next assembly quantities can be observed directly by counting the arrows on each connecting line. Total requirements cannot be observed directly but can be deduced.

Development of the Mathematical Model[1]

The basic information required for production control is usually listed on so-called assembly parts lists. An example, shown on Fig. 1, refers to a Panel with part number 435090012. This "Makes Assembly" is made up of seven different articles as shown on the parts list under the heading N. A. QTY (Next Assemblies Quantity). It is shown how many of these articles are needed. Thus, the bushing part number 420990309 is required in a quantity of three for each of the panels 435090012.

There would be a similar sheet for every assembly, and in our hypothetical factory there would be perhaps a few thousand of these sheets. Our first problem, then, is to put this information into concise mathematical form. Before we do this it will be useful to think of this problem in terms of a graphical representation. Fig. 2

[1] For a more complete description, see A. Vazsonyi, "The use of mathematics in production and inventory control," *Management Sci.*, vol. 1, pp. 70–85; October, 1954, and another article by A. Vazsonyi to be published in No. 3 of the same journal.

	1	2	3	4	5	6	7	8	9
1	0	0	0	2	0	0	1	1	0
2	0	0	0	0	0	0	0	0	0
3	1	0	0	0	3	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	2	0	0	0	0	0	2	1	0
6	0	2	0	0	1	0	0	0	3
7	0	1	0	1	0	0	0	0	0
8	0	2	0	0	0	0	0	0	1
9	0	0	0	0	0	0	0	0	0

Fig. 3—Next assembly quantity table. This is a concise mathematical representation of the information contained in the assembly parts lists.

	1	2	3	4	5	6	7	8	9
1	1	3	0	3	0	0	1	1	1
2	0	1	0	0	0	0	0	0	0
3	7	33	1	27	3	0	13	10	10
4	0	0	0	1	0	0	0	0	0
5	2	10	0	8	1	0	4	3	3
6	2	12	0	8	1	1	4	3	6
7	0	1	0	1	0	0	1	0	0
8	0	2	0	0	0	0	0	1	1
9	0	0	0	0	0	0	0	0	1

Fig. 4—Total requirement factor table. Observe, say, the *second* column relating to A_2. The *third* element from the top in this column relates to A_3 and displays the number 33. This means that 33 A_3's are required (in total) for each A_2. This can be confirmed by a direct count from Fig. 10.

shows a highly simplified situation when there are only three top assemblies and a total of nine articles. The Next Assembly Quantities are shown by the arrows on the figure. The various Next Assembly Quantities shown on Fig. 2 are represented in a tabular form in Fig. 3, and this Next Assembly Quantity matrix is a concise mathematical representation of the information contained in the assembly parts list.[2] For instance, it can be seen either from Fig. 2 or Fig. 3 that each Article 7 or A_7 takes two A_5's directly. We use the word "directly" advisedly as A_7 requires, in total, four of A_5, as A_7 requires two A_1's directly and each A_1, in its turn, requires two of A_5 directly. We are, of course, interested to know how many of each article is required directly or indirectly for each other article. This information cannot be read directly from the Next Assembly Quantity table, and our problem is how to determine these total requirement factors from the Next Assembly Quantity table.

In the same way as we put the Next Assembly Quantities into tabular form, we put the Total Requirement Factors into a table. To illustrate the case, Fig. 4 shows the Total Requirement Factor table associated with the illustrative example in Fig. 2.[3] We agree that each A_7 takes four of A_5's in total. In Fig. 4, in the fifth row, under the seventh column, the number 4 is listed. The figure shows that 33 of A_3 is required for each A_2. Verification of this in Fig. 2 requires a careful tracing of the various arrows. This tracing in Fig. 2 can be described by the following statement:

Total number of A_5's required for each A_2 =
(Number of A_5's going directly into each A_1)
·(Total number of A_1's required for each A_2)
+(Number of A_5's going directly into each A_7)
·(Total number of A_7's required for each A_2)
+(Number of A_5's going directly into each A_8)
·(Total number of A_8's required for each A_2).

Note again, carefully, the distinction between the statement "total number of A's required. . ." and "number of A's going directly into. . . . "

We proceed now to put the above statement into mathematical form. We denote by $N_{i,p}$ the Next Assembly Quantity, which indicates the number of A_i's going directly into an A_p. Furthermore, we denote by $T_{p,j}$ the Total Requirement Factor, that is, the total number of A_p's required for each A_j. Then the statement above can be written as

$$T_{5,2} = N_{5,1}\cdot T_{1,2} + N_{5,7}\cdot T_{7,2} + N_{5,8}\cdot T_{8,2}, \qquad (1)$$

or

$$T_{5,2} = \sum_p N_{5,p}T_{p,2}.$$

A graphical representation of these two equations is given in Fig. 5.

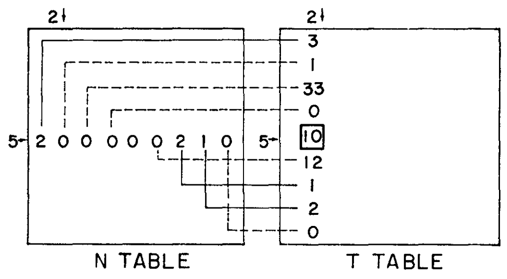


Fig. 5—Schematic representation of the equation

$$T_{5,2} = \sum_p N_{5,p}T_{p,2}.$$

The second number of the fifth row of the T-Table equals the "scalar multiple" of the fifth row of the N-Table and second column of the T-Table:

$$10 = 2\times 3 + 0\times 1 + 0\times 33 + 0\times 0 + 0\times 10 + 0 \times 12 + 2\times 1 + 1\times 2 + 0\times 0.$$

It is quite plausible now to conclude that the above formula can be generalized for any pair of articles A_i and A_j. Therefore, we deduce that

$$T_{i,j} = \sum_p N_{i,p}\cdot T_{p,j} \qquad i \neq j. \qquad (3)$$

We recall now our convention that

$$N_{i,i} = 0 \qquad (4)$$

$$T_{i,i} = 1 \qquad (5)$$

[2] Note that we put 0's in the diagonal. This makes the mathematical development easier.

[3] Note that we put 1's in the diagonal. This again makes the mathematical development easier.

These relations can be written in a more concise form by using matrix algebra:

$$[T] = \left[\frac{I}{I - N}\right], \tag{6}$$

where $[I]$ denotes the unit matrix. Our first problem, the determination of the Total Requirement Factors, leads to the problem of matrix inversion as shown by (6). Or, to put it in another way, the problem is to solve the system of (3) and (5) for the unknown T's. If these matrices had no special property, it would be practically impossible to carry through this computational task when there are thousands of articles. Fortunately, most of the elements of the matrices are zeros and the matrix is "triangular"; consequently, special procedures can be developed. Before we describe the computational technique, we develop our mathematical model further.

As we said in the introduction, we need to know the total requirements for each part as related to a shipping schedule. Suppose we are shipping only articles A_2, A_4, A_9, and A_5, the last being a spare. How do we compute the quantity of A_5's required? Clearly

Quantity of A_5's required =
(Total number of A_5's required for each A_2)
×(shipping requirements of A_2)
+(Total number of A_5's required for each A_4)
×(shipping requirement of A_4)
+(Total number of A_5's required for each A_9)
×(shipping requirement of A_9)
+(Shipping requirement of A_5).

In order to put this in mathematical form, we introduce the notation that the shipping requirements are given by $S_1, S_2, \cdots$ and that the unknown requirements for the articles are $X_1, X_2, \cdots$ With this notation the above verbal statement can be written as

$$X_5 = T_{5,2}\cdot S_2 + T_{5,4}\cdot S_4 + T_{5,9}\cdot S_9 + S_5. \tag{7}$$

This equation can again be written as

$$X_5 = \sum_k T_{5,k}\cdot S_k. \tag{8}$$

Again, it is plausible to generalize for any article A_i

$$X_i = \sum_k T_{i,k}\cdot S_k. \tag{9}$$

This last equation then gives a method of determining the quantity of each article required once the T matrix is computed and if the shipping schedule is given. It is important from the computational point of view to recognize that most of the S's are zero, as usually only a small fraction of the articles manufactured are shippable.

The Problem of Scheduling

So far, we have concerned ourselves only with the problem of quantities required. Now we propose to introduce the time element. We will assume that our hypothetical factory operates in production periods, and we will assume that the shipping requirement is given for each article in each period. The question we must answer now is how many of each article is to be made in each period. We denote by s_i^m the number of A_i's to be shipped in the mth period. Similarly, we denote the unknown quantity of A_i's to be manufactured in the mth period by x_i^m.

In every manufacturing process, articles must be made in a certain technological sequence. Looking at Fig. 2, one can see that, say, article A_1 must be made before A_8, as the former is a part of the latter. It is the usual practice in production control to make up so-called setback charts. Fig. 6 illustrates the setback chart for top assembly A_9. Each article is allowed a cer-

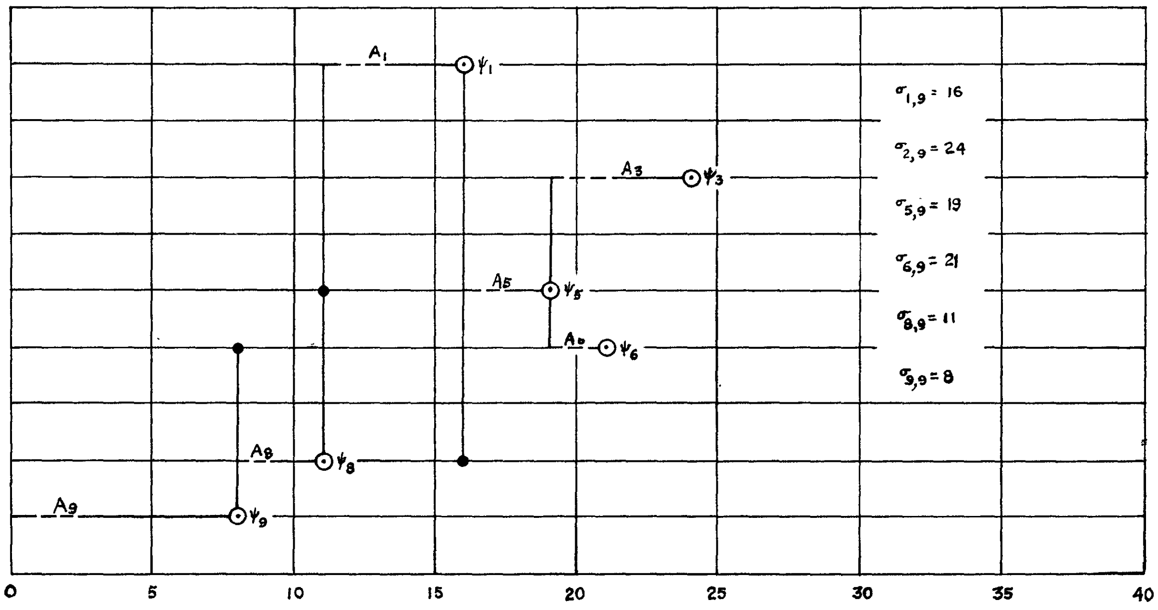


Fig. 6—Setback chart in Assembly A_9. As an example, the setback of A_1 is 16.

tain make-time, and then a safety cushion is introduced. For instance, in Fig. 6, Article A_3 is allowed three production periods, and a cushion of two production periods is introduced for safety. It can be seen from the figure, for instance, that A_5 is to be started 19 periods earlier than A_9 is to be shipped. Suppose we know that s_9^{30} is given indicating the number of A_9's to be shipped in the 30th production period. Then we have

$$x_{5,9}^{11} = 3s_9^{30} \tag{10}$$

where the left-hand side indicates the number of A_5's to be made in the 11th production period, provided the shipping requirements of A_9 are the only requirements to be taken into consideration. The factor 3 appears as each A_9 takes three A_5's. Eq. (10) easily generalizes into

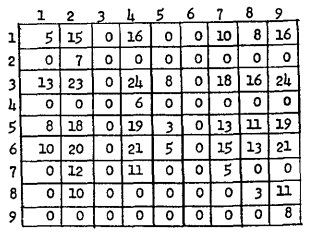
$$x_{5,9}^{m} = 3s_9^{m+19}, \tag{11}$$

or into

$$x_{5,9}^{m} = T_{5,9}s_9^{m+\sigma_5}, \tag{12}$$

where $\sigma_{5,9}$ denotes the setback of Article A_5 "in" Article A_9.

In order to generalize this relationship to any article "in" any other article, we introduce the concept of the setback matrix $\sigma_{i,k}$, as illustrated in Fig. 7. With this



	1	2	3	4	5	6	7	8	9
1	5	15	0	16	0	0	10	8	16
2	0	7	0	0	0	0	0	0	0
3	13	23	0	24	8	0	18	16	24
4	0	0	0	6	0	0	0	0	0
5	8	18	0	19	3	0	13	11	19
6	10	20	0	21	5	0	15	13	21
7	0	12	0	11	0	0	5	0	0
8	0	10	0	0	0	0	0	3	11
9	0	0	0	0	0	0	0	0	8

Fig. 7—Setback matrix $[\sigma]$. Each column represents a setback chart. For instance, the ninth column represents the setback chart shown in Fig. 6.

notation we get a general formula for the number of A_i's to be made in production period m, with the proviso that these A_i's are earmarked to a shippable A_i:

$$x_{i,k}^{m} = T_{i,k}s_k^{m+\sigma_{i,k}}. \tag{13}$$

If we want to determine the total number of A_i's to be made in the mth period, we have to add up the various A_i's required for each shippable article, and so we get

$$x_i^{m} = \sum_k x_{i,}{}^{m}{}_{k} \tag{14}$$

or

$$x_i^{m} = \sum_k T_{i,k}s_k^{m+\sigma_{i,k}}. \tag{15}$$

It can be seen then that this last equation allows us to compute requirements as imposed by shipping schedules.

Machine Loading

Finally, we proceed to the machine hours computations associated with the schedule. We assume that each article goes through a number of machines and that the standard times (or whatever times we want to compute) are given. This information is visualized as a $\tau_{n,i}$ matrix where $\tau_{n,i}$ denotes the number of hours required on machine type n when article A_i is manufactured. The total labor hours required in production period m on machine type n is given by

$$h_n^{m} = \sum_i \tau_{n,i}x_i^{m}. \tag{16}$$

If it is desired to compute the labor hours which are associated with the manufacture of a shippable article A_k, we use the formula

$$h_{n,k}^{m} = \sum_i \tau_{n,i}x_{i,k}^{m}. \tag{17}$$

In these formulae we neglected set-up time. If this is important, the formula can be modified by adding these set-up times. Man-hour computations for assembly or other type of labor can be computed in a very similar fashion.

Computational Procedures

Computation of the Total Requirement Factor Matrix [T]

Eq. (3) describes the computation required to generate [T]. At first glance it appears that matrix inversion is involved, i.e., the system of equations for the [T] matrix must be solved. However, due to the "triangular" nature of the matrix involved, it is possible to consider the elements of [T] in such a manner that as each row is developed, this row and previously developed rows can be used to develop the next row. This amounts to working downward from the top of the Gozinto graph. The computation proceeds as follows:

1. Store [N] in a manner such that it can be consulted one row at a time.
2. Store [T] by columns, initially representing all the elements as "unknown."
3. Insert unity in the elements of the diagonal of [T].
4. Search the rows of [N] for those which contain all zero elements. These are A_i which go into no A_p, and are therefore "top assemblies," or at the top of the Gozinto graph. The elements $T_{i,j}$ are developed by multiplying the elements of row i of [N] by the elements of column j of [T], and summing the products. Since all of the elements of the row of [N] are zero, all elements of the corresponding row of [T] are zero (except the diagonal). Thus, the corresponding pth elements of the columns of [T] can be filled in with zeros. In our example, we have now filled in the diagonal and rows 2, 4, and 9 of [T], as shown in Fig. 8. Record which rows p of [T] are now known.
5. Search the rows of [N] to find those which contain nonzero elements only in columns p which correspond to the known rows p of [T]. This locates the next lower echelon of articles on the Gozinto graph. In the example, this search of [N] would be conducted watching for

$$\begin{array}{c|ccccccccc}
 & & & & & j & & & & \\
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\hline
1 & 1 & x & x & x & x & x & x & x & x \\
2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & x & x & 1 & x & x & x & x & x & x \\
4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
i\ p\ 5 & x & x & x & x & 1 & x & x & x & x \\
6 & x & x & x & x & x & 1 & x & x & x \\
7 & x & x & x & x & x & x & 1 & x & x \\
8 & x & x & x & x & x & x & x & 1 & x \\
9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}$$

Fig. 8—First steps in developing $[T]$.

nonzero elements only in columns 2, 4, and 9. This condition would be satisfied for rows 7 and 8 of $[N]$. When a row of $[N]$ is found which satisfies this condition, hold it and multiply it successively with all the columns of $[T]$. As each element of $[T]$ is developed, insert it in the column of $[T]$. The multiplication of the 7th row of $[N]$ by the second column of $[T]$ is diagrammed in Fig. 9. Note that wherever an unknown appears in $[T]$, the corresponding element of $[N]$ is zero. This will always be true in this procedure. Add these new known j's to the list of known rows of $[T]$.

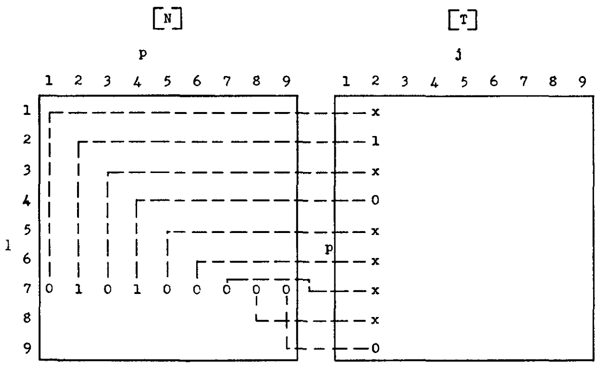


Fig. 9—Computation of $T_{7,2}$.

$$\begin{aligned}
T_{7,2} &= \sum_p N_{7,p}T_{p,2} \\
&= 0\cdot x + 1\cdot 1 + 0\cdot x + 1\cdot 0 + 0\cdot x + 0\cdot x + 0\cdot x + 0\cdot x + 0\cdot 0 \\
&= 1.
\end{aligned}$$

6. As the elements of the rows of $[T]$ are developed and placed in the proper positions in the columns of $[T]$, save those elements of each row of $[T]$ which fall in columns k corresponding to shippable articles A_k, and write these out, as a row, before proceeding. The reason for this is that we need the $T_{i,k}$ stored in rows for the following computation, and since we are developing them in rows during the present task, we may as well write out the rows and save some of the sorting that would be entailed in changing $[T]$ from column form to row form.

7. With the pass described in steps 5 and 6 complete, repeat the search described in step 5, now matching nonzero elements of the rows of $[N]$ against the augmented list of known rows of $[T]$. This will locate the next lower echelon of articles on the Gozinto graph, make it possible to compute additional rows of $[T]$, etc. This iteration is continued until all the elements of $[T]$ are computed.

8. Sort the rows of $[T]$, written out in step 6, into ascending sequence on the row number i.

Computation of the Production Requirements Schedule, x_i^m

Eq. (13) shows that the ingredients for this computation are the Total Requirements Factor matrix $[T]$, the setback matrix $[\sigma]$, and the shipping schedules s_k^m. The procedure is as follows:

1. Bring in the first row of $[T]$, and find the first nonzero element in that row; the co-ordinate k represents an article A_k into which article A_1 goes. In the example given in Fig. 10, the first A_k is A_2.

$$[T]\quad \begin{array}{c|ccc}
 & & k & \\
 & 2 & 4 & 9 \\
\hline
1 & 3 & 3 & 1 \\
2 & 1 & 0 & 0 \\
3 & 33 & 27 & 10 \\
4 & 0 & 1 & 0 \\
i\ 5 & 10 & 8 & 3 \\
6 & 12 & 8 & 6 \\
7 & 1 & 1 & 0 \\
8 & 2 & 0 & 1 \\
9 & 0 & 0 & 1
\end{array}
\qquad
[\sigma]\quad \begin{array}{c|ccc}
 & & k & \\
 & 2 & 4 & 9 \\
\hline
1 & 3 & 3 & 3 \\
2 & 1 & - & - \\
3 & 5 & 5 & 5 \\
4 & - & 1 & - \\
i\ 5 & 4 & 4 & 4 \\
6 & 4 & 4 & 4 \\
7 & 2 & 2 & - \\
8 & 2 & - & 2 \\
9 & - & - & 2
\end{array}$$

Fig. 10—The T-matrix and σ matrix, including columns k only for shippable articles A_k.

2. Bring in the first row of $[\sigma]$. This will have nonzero elements in the same positions as in $[T]$, since wherever an article A_i goes into an article A_k there must be a setback stated. Look up the setback $\sigma_{i,k}$ corresponding to the element found in step 1. In our example, this is $\sigma_{1,2}=3$.

3. Bring in the shipping schedule s_k^m associated with the article A_k found in step 1. If the first production period for which we wish to compute the schedule on A_1 is period m_1, form the sum $(m_1+\sigma_{1,k})$. This is the period of the shipping schedule in which we are interested. Pull the quantity $s_k^{m_1+\sigma_{i,k}})$ found in this period, and multiply it by $T_{i,k}$. In our example, we compute period 16 first, which means we look up $s_2^{19}=1$ (Fig. 11), and form the product $T_{1,2}s_2^{19}=3\times 1=3$.

		m																				
		16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
	s_2	0	0	4	1	7	6	6	2	3	1	1	0	0	0	1	3	0	0	0	0	0
k	s_4	4	4	5	4	3	3	2	1	0	1	2	0	0	0	0	0	0	0	0	0	0
	s_9	3	1	2	4	9	0	7	1	3	2	4	6	6	1	0	0	2	1	8	0	1

Fig. 11—Shipping schedules for articles A_k.

4. Find the next nonzero element in the first row of $[T]$. The example shows this as $T_{1,4}=3$.

5. Look up the corresponding setback in the first row of $[\sigma]$. In the example this is $\sigma_{1,4}=3$.

6. Bring in the shipping schedule for the new A_k found in step 4. Here, we would bring in schedule $s_4{}^m$. In period (16+3) we find $s_4{}^{19}=4$. Form the product $T_{1,4}s_4{}^{19}=3\times 4=12$, and add this to the product developed in step 3.

7. Continue this iteration until all of the nonzero elements of the first row of $[T]$ have been considered. This will form the first element of $x_i{}^m$. In the example, this forms

$$x_1{}^{16} = T_{1,2}s_2{}^{19} + T_{1,4}s_4{}^{19} + T_{1,9}s_9{}^{19}$$
$$= 3 \times 1 + 3 \times 4 + 1 \times 4 = 19.$$

At this point, all of the shipping schedules needed for computing the entire production schedule for A_1 have been assembled.

8. Repeat the above steps for the next production period; in our case, repeat for production period 17. Note that the same elements of $[T]$ and $[\sigma]$ are used, and that the references to the shipping schedules are made one production period further along than was the case in steps 6 and 7.

9. Continue the iteration of steps 1 through 8 until the production schedule for A_1 has been computed to the period m representing the planning horizon. This completes the computation for article A_1. Write this schedule out.

10. Bring in the second row of $[T]$, the second row of $[\sigma]$, etc., repeating the procedures described above, starting in step 1, and developing the production schedule for article A_2. Repeat all this until the production schedules for all A_i have been completed.

Computation of the Machine Loading Schedules, $h_n{}^m$

The computation suggested by (16) is the same as a straightforward matrix multiplication, considering the production schedules $x_i{}^m$ as a matrix of dimensions i, m. This suggests storing $[\tau]$ by rows and $x_i{}^m$ by columns. For several reasons, however, these machine loading schedules will be computed in a slightly different manner. The first of these reasons is that the production schedules are developed in rows (see above), are printed out in this form, and are extremely lengthy. The sorting job to be done in changing these schedules to column form is an enormous task in many practical examples. Thus, we will suggest here a scheme which makes use of each element of $x_i{}^m$ as it appears in row sequence. The second reason for adopting the computing scheme to be suggested here is that the contents of $[\tau]$ are available in column form in the lists used by production people. Thus, if we can use $[\tau]$ in columns, rather than sort the elements into rows, we will save some effort. The third reason we consider this scheme is that the dimensions of the $h_n{}^m$ "matrix" are reasonably small, as will be seen in a later section of this paper. These dimensions are such that in many problems the entire matrix can be stored in an internal memory of 1,000 to 6,000 words. The method is as follows.

1. Assign space in the internal memory for a matrix of elements with n (max) rows and m (max) columns, all elements containing the initial value zero.

2. Bring in the first column of $[\tau]$.

3. Bring in the first schedule, or "row," of $x_i{}^m$.

4. Multiply the first element of the first row of $x_i{}^m$, i.e., $x_1{}^1$, by the elements of the first column of $[\tau]$, i.e., $\tau_{n,1}$, adding each product into the $h_n{}^m$ matrix at the appropriate address of the first column, n,1. This is demonstrated in the example of Fig. 12, where $x_1{}^1$ is seen to

$$\overset{[\tau]}{\begin{array}{c|ccc|} & i_1 & i_2 & i_3 \\ \hline n_1 & \tau_{11} & \tau_{12} & \tau_{13} \\ n_2 & \tau_{21} & \tau_{22} & \tau_{23} \\ \hline \end{array}} \quad \times \quad \overset{x_i^m}{\begin{array}{c|ccc|} & m_1 & m_2 & m_3 \\ \hline i_1 & x_1^1 & x_1^2 & x_1^3 \\ i_2 & x_2^1 & x_2^2 & x_2^3 \\ i_3 & x_3^1 & x_3^2 & x_3^3 \\ \hline \end{array}}$$

$$= \overset{h_n^m}{\begin{array}{c|ccc|} & m_1 & m_2 & m_3 \\ \hline n_1 & (\tau_{11}x_1^1+\tau_{12}x_2^1+\tau_{13}x_3^1) & (\tau_{11}x_1^2+\tau_{12}x_2^2+\tau_{13}x_3^2) & (\tau_{11}x_1^3+\tau_{12}x_2^3+\tau_{13}x_3^3) \\ n_2 & (\tau_{21}x_1^1+\tau_{22}x_2^1+\tau_{23}x_3^1) & (\tau_{21}x_1^2+\tau_{22}x_2^2+\tau_{23}x_3^2) & (\tau_{21}x_1^3+\tau_{22}x_2^3+\tau_{23}x_3^3) \\ \hline \end{array}}$$

Fig. 12—Example of the product
$$h_n{}^m = \sum_i \tau_{n,i}x_i{}^m.$$

appear only in the elements of the first column of $h_n{}^m$, multiplied successively by the elements of the first column of $[\tau]$.

5. Repeat step 4, multiplying the second element of the first row of $x_i{}^m$, i.e., $x_1{}^2$, by the same first column of $[\tau]$, i.e., $\tau_{n,1}$, adding each product into the $h_n{}^m$ matrix at the appropriate address in the second column. Continue this procedure until all the elements of the first row of $x_i{}^m$ have been considered.

6. Bring in the next column of $[\tau]$ and the next row of $x_i{}^m$, and repeat the above procedure.

7. Repeat the above iteration until all columns of $[\tau]$ and all rows of $x_i{}^m$ have been exhausted. Read out the schedules $h_n{}^m$ as rows of the matrix which have been developed in the internal storage.

For the computation of labor requirements, a matrix of the same type as $[\tau]$ is provided, where the elements state the total number of man-hours of type n required to construct one article A_i. The n dimension of such a labor requirements matrix is the same order of magnitude as the n dimension of the machine-hour requirements matrix. The computation would be carried out in the same manner as described here for machine loading.

Verification Procedures

Several procedures for verifying accuracy will be included in the present exercise. The checks included here will serve to indicate the type of check which can be performed, and their inclusion will make the time and space estimates more practical.

Input Checks

To maintain a check on the introduction of changes in any of the schedules and matrices, we will include checks to monitor the accuracy of the steps performed in the input translations. The purpose of these checks will be to detect errors in the initial keyboard entry, translation of part numbers, and actual modification of the data. These checks require that certain total quantities be generated by the personnel initiating the changes, so that the machine procedures will have something to check against. Briefly, the quantities required and procedures to be followed are as follows:

Changes in [N].—Parts lists changes are to be accompanied by a new total number of parts represented by the assembly parts list. Note that the total number of parts in the assembly parts list for an assembly A_p is represented by the sum of the elements of the column for A_p in [N]. The sums of all columns will be carried with the stored [N], forming an additional row. When modifications of [N] are made, each column will be checked to insure that the sum of the new set of elements agrees with the new total which was supplied along with the changes.

Changes in [σ].—When the setback structure is changed by the production department, the new set of setbacks for a given A_k is to be accompanied by a simple sum of the quantities stating the setbacks. When the changes have been recorded in the stored form of [σ], the sum of the columns will be compared with these totals, in the same manner as [N] is checked.

Changes in s_k^m.—For changes in the shipping schedules, it will be assumed that checks on the inputs will be performed as part of other (probably daily) procedures, and hence this group of input checks will not be considered here. (These include checks on the changes caused by new orders, order changes, and shipments of completed assemblies.) It will be assumed that the total A_k on each order and the cumulative shipping schedules are stored and are available for subsequent checks.

Changes in [τ].—Here total machine hours (of all types) for each A_k should be supplied. Each column of [τ] will be compared with the new totals supplied with the changes.

Checks on Accuracy of Computation

The check which will be used repeatedly makes use of a check column or check row, or both, in the matrices and schedules employed here. Appendix I describes the principle employed.

Check on Computation [T].—Here we will augment [N] with an additional row. The contents of this row are simply the sum of the number of parts going directly into each A_p. (Note that these are the same quantities used to verify the accuracy of changes introduced into [N].) This will generate an augmented total requirements matrix [T^*]. The elements of the last row of [T^*] must be the sum of the elements of the column above each. This check will be made when [T^*] is complete. A run which reads all of the rows of [T^*] will be made, adding each element of a given row into an address determined by the column k. By the time the last (check) row of [T^*] is reached, the sum of the elements of each column of [T] will be complete in the storage, and these sums can be compared with the elements of the last row of [T^*].

Check on Computation of x_i^m.—The principle of Appendix I can be applied here in a slightly modified form. The checking technique must be changcd a little to take into account the fact that each series of products, developing a schedule for one A_i, does not make use of the entire contents of the schedules s_k^m; a section of each schedule is used, starting from the m representing the setback for A_i going into that A_k, and ending at a period equal to the last production period computed plus the setback. Cumulative schedules S_k^m will be stored. As the x_i^m schedule is computed for one A_i, the difference between the cumulative quantities at the beginning and end of the section of the shipping schedule will be taken. This will be done for all the A_k involved, producing in effect an extra last "column" of s_k^m. This "check column," multiplied by the rows of [T], will produce a check column in the resulting x_i^m. The elements in this latter column must be the sum of the elements of each schedule x_i^m for one A_i.

Check on Computation of h_n^m.—Here a more complete check will be carried out. The matrix [τ] will be augmented with an extra row containing the sum of the machine-hours of all types n required to make each A_i. Note that this is a quantity which can be sypplied by the people developoing the production techniques, and this is the quantity used in checking the accuracy of changes put into [τ]. The "matrix" [x] has an augmenting "column" derived when the x_i^m schedules are derived. The resulting augmented h_n^{m*} will contain an extra row and column which will contain the sums of the appropriate rows and columns of h_n^m. With horizontal and vertical checks, an error in one of the elements should be accompanied by check failures in one of the horizontal and one of the vertical checks. These two failures give the co-ordinates of the incorrect element, making it possible to program for a recomputation of that element alone.

Periodic Data-Processing Procedure

With the background of the preceding sections, we can now outline the procedure to be performed each time the production schedule, machine load requirements, and labor requirements are to be developed.

1. Introduce the changes in [N]. Working from the assembly parts lists and check sums, make a run through the rows of [N] replacing elements with new quantities, introducing new assemblies and deleting others. This reflects the changes which have taken place in the company products since the last computation. Perform a second run through [N], checking the control sums.

2. Compute the total requirements factor matrix [T], as outlined above in the section on computation. This produces a complete matrix stored in columns, and a reduced version containing only the columns for shippa-

ble articles A_k, stored in row form. Verify the reduced version of the matrix by the method described in the preceding section.

3. Sort the rows of $[T]$ computed in step 2 into ascending sequence on i, so that they will be available in the proper sequence for performing the production requirements computation.

4. Introduce the changes in $[\sigma]$. As products are added or dropped, as engineering or production technique changes are made, and as experience is gained, the values of the setbacks will change. The changes are introduced and verified in the same manner as the changes in $[N]$. Since the changes in $[\sigma]$ may well be developed by different personnel than those developing the changes in $[N]$ (and hence, $[T]$), it is entirely possible that inconsistent changes will be introduced into $[N]$ and $[\sigma]$. For example, the affects of a new assembly may pass through the paper mill leading to the changes in $[N]$ more rapidly than changes go through the mill leading to changes in $[\sigma]$. Thus, when the computation of production schedules is carried out, certain elements of $[T]$ may find no corresponding elements in $[\sigma]$. If it is not convenient to check these two sets of changes outside of the data processor, it is strongly recommended that a special consistency check run be made in the data-processing procedures. This run might compare the two sets of inputs, or, better, compare $[T]$ and $[\sigma]$ to make sure that each has nonzero elements in the same locations.

5. Consolidate shipping schedules. The organization of the actual schedules may be somewhat different from that required for s_k^m. For the purposes of this computation, we want all orders for article A_k to be consolidated into one schedule. The actual schedules may be broken into several schedules for each A_k, the separate schedules being for different customers, or different destinations, or both. For this exercise, assume that such is the case, and that it is necessary to perform a consolidation of several schedules to produce the desired s_k^m.

6. Compute the production requirements schedules, x_i^m, as outlined above in the section on computation.

7. Print out the production requirements schedules. The schedules developed in the computation of step 6 are in a suitable sequence for print-out. Some editing will be necessary to arrange a suitable format.

8. Introduce the changes in $[\tau]$. The modification and checking procedure for making changes in the machine-hour requirements matrix is similar to that described above for $[N]$ and $[\sigma]$. The program will be somewhat different here, because $[\tau]$ is stored in columns, while the $[N]$ and $[\sigma]$ are stored in rows.

9. Compute the machine loading schedules, h_n^m, using the general program stated above under computation.

10. Print out the machine loading schedules. The schedules developed in step 9 are in suitable sequence for print-out. Some editing will be necessary to arrange a suitable format.

11. Introduce the changes in the matrix of type $[\tau]$ representing labor man-hour requirements, as in step 8.

12. Compute the labor loading schedules, as in step 9.

13. Print out the labor loading schedules, as in step 10.

Magnitude of the Problem

The size and content of the various matrices and schedules involved in this exercise may be measured by the quantities defined in Table I. The range of values given in this table serve to illustrate the size of problems encountered in practice and which appear to be of sufficient magnitude to call for the services of high-speed data-processing equipment. A general picture of the range of dimensions of the several matrices and schedules is given in Table II.

Table III gives the parameters for three production control examples which have been encountered by the authors.

Consider the storage space required for storing the matrix $[N]$.

Case 1

If all elements of $[N]$ are stored, if actual part numbers are used (often requiring the storage of characters other than digits), if each row is stored by listing i once at the beginning of each row, identifying the j by the position of each element in the row, then the number of digits and characters stored is

$$(Q_1)^2 D_N \text{ (digits)} + Q_1 C_p \text{ (characters)}.$$

Digits and characters are separated here so that they may be translated into actual storage space in different types of processors. One processor may be constructed so that it can handle alphabetic characters and other symbols internally, with all digits and characters occupying the same amount of storage space; another processor will require the use of two digit spaces for storing one alphabetic character. The storage, in digits and characters, required for $[N]$ in the three examples is shown in the first row of Table IV. Since this represents the order of 20,000 feet, five million feet, and fourteen million feet of magnetic tape, respectively (see below for the assumptions made concerning magnetic tape storage density), it appears to be advisable to consider means for reducing the storage required.

Case 2

A considerable saving in storage space can be effected if we take advantage of the fact that most of the elements of $[N]$ are zero. Consider now the storage space required if we store only nonzero elements. We will store i at the beginning of a row, and store j with each element. The storage required is

$$Q_1 q_1 (D_N \text{ digits} + C_p \text{ characters}) + Q_1 C_p \text{ characters}.$$

The resulting storage for our examples is given in the second row of Table IV.

TABLE I

SUMMARY OF QUANTITIES REQUIRED TO MEASURE THE SIZE OF THE COMPUTATION TASK

	Range of values
Q_1 = number of different A_i. This gives the dimensions of $[N]$, the vertical dimension of $[T]$, $[\sigma]$ and x_i^m, and the horizontal dimension of $[\tau]$.	$1,000 \leqq Q_1 \leqq 100,000$
Q_2 = number of shippable articles A_k. This gives the horizontal dimension of $[T]$ and $[\sigma]$, and the vertical dimensions of s_k^m.	$100 \leqq Q_2 \leqq 1,000$
Q_3 = average number of periods represented in shipping schedules. This gives the average horizontal dimension of s_k^m.	$10 \leqq Q_3 < 1,000$
Q_4 = number of periods into the future the production and machine load schedules are to be computed. This gives the horizontal dimension of x_k^m and h_n^m.	$5 \leqq Q_3 < 1,000$
Q_5 = number of different types of machine-hours (n). This gives the vertical dimension of $[\tau]$ and h_n^m.	$10 \leqq Q_4 < 1,000$
q_1 = average number of different A_i required to assemble an A_p. This is the average length of assembly parts lists, and gives the number of nonzero elements in a column of $[N]$.	$5 \leqq q_1 \leqq 100$
q_2 = average number of different A_i appearing in assemblies of all types and at all levels of the Gozinto graph. This is the average number of nonzero elements in a column of the $T_{p,i}$ matrix. This quantity will always be larger than q_1. It depends on the number of different assemblies a given part or assembly goes into, and upon the number of different levels in the Gozinto graph.	$10 \leqq q_2 \leqq 100$
q_3 = average number of different A_i appearing in shippable articles A_k. This gives the average number of nonzero elements in a column of $[T]$ and $[\sigma]$.	$10 \leqq q_3 \leqq 10,000$
q_4 = average number of different types of machine-hours required to make an assembly A_i. This gives the average number of nonzero elements in a column of $[\tau]$.	$5 \leqq q_4 \leqq 100$
q_5 = the number of levels in the Gozinto graph.	$2 \leqq q_5 \leqq 100$
$N_{i,p}$ = number of A_i required to make one A_p.	$0 \leqq N_{i,p} < 1,000$
D_N = number of digits required to store one element of $[N]$.	$2 \leqq D_N \leqq 3$
$T_{i,k}$ = total number of A_i required to make one shippable article A_k.	$0 \leqq T_{i,k} < 10,000$
D_T = number of digits required to store one element of $[T]$.	$2 \leqq D_T \leqq 4$
s_k^m = number of A_k to be shipped in production period m.	$0 \leqq s_k^m < 100,000$
D_m = number of digits required to store the designation of a period.	$3 \leqq D_m \leqq 4$
S_k^m = cumulative number of A_k to be shipped by production period m.	$0 \leqq S_k^m < 1,000,000$
$\sigma_{i,k}$ = the total setback of an A_i going into A_k.	$0 \leqq \sigma_{i,k} < 10,000$
x_i^m = the number of A_i to be started in production period m.	$0 \leqq x_i^m < 100,000$
$\tau_{n,i}$ = the total number of machine-hours of type (n) required to make one A_i.	$.00 \leqq \tau_{n,i} < 100.00$
h_n^m = the total number of machine-hours of type n required in production period m.	$0 \leqq h_n^m \leqq 10,000$
C_p = number of characters required to store a standard part number.	$5 \leqq C_p \leqq 24$
D_p = number of digits required to store a translated part number used within the data processor.	$3 \leqq D_p \leqq 5$
$D_{(\ )ch}$ = number of digits required to store a check sum for one column or row of matrix ( ).	$4 \leqq D_{(\ )ch} \leqq 7$
C_n = number of characters required to store a standard machine-type designation n.	$5 \leqq C_n \leqq 10$
D_n = number of digits required to store a translated machine-type designation n used within the data processor.	$2 \leqq D_n \leqq 3$

TABLE II

RANGE OF MAGNITUDES OF THE MATRICES AND SCHEDULES

Matrix or Schedule	Dimensions	
	Minimum	Maximum
$[N]$	1,000×1,000	100,000×100,000
$[T]$, $[\sigma]$	1,000× 100	100,000× 1,000
s_k^m	100× 10	1,000× 1,000
x_i^m	1,000× 5	100,000× 1,000
$[\tau]$	10×1,000	1,000×100,000
h_n^m	10× 5	1,000× 1,000

While these requirements are more reasonable than in Case 1, it still appears to be highly desirable to reduce the storage requirements further, if possible. The example considered here, $[N]$, is but one of the several matrices and schedules involved in this problem. Any small per cent reduction in storage requirements will buy a very real saving in storage medium cost.

Case 3

Consider now the use of two sets of part numbers. A set of part numbers will be established for internal use throughout the computation. A translating table must be carried, giving the part numbers used by people and the corresponding part numbers used internally. Now, $[N]$ requires

$$M_n = Q_1 q_1 (D_N + D_p) + Q_1 D_p \text{ digits.} \qquad (18)$$

The translating table occupies the following storage:

$$M_{Tr1} = Q_1(C_p \text{ characters} + D_p \text{ digits}). \qquad (19)$$

TABLE III
EXAMPLES OF PRODUCTION CONTROL PARAMETERS

	A	B	C
$Q_1=$	3,000	50,000	80,000
$Q_2=$	100	250	200
$Q_3=$	35 weeks	18 months	36 months
$Q_4=$	25 weeks	12 months	24 months
$Q_5=$	50	100	250
$q_1=$	10	15	20
$q_2=$	20	20	30
$q_3=$	50	75	2,500
$q_4=$	5	10	15
$q_5=$	7	20	50
$D_N=$	2	2	3
$D_T=$	2	3	4
$D_s=$	4	4	3
$D_S=$	5	5	4
$D_m=$	3	3	3
$D_\sigma=$	2	2	2
$D_x=$	4	4	4
$D_\tau=$	2	2	3
$D_h=$	3	4	4
$C_p=$	9	14	14
$D_p=$	4	5	5
$D_n=$	2	3	3
$C_n=$	5	6	8

The figures in row 3 of Table IV show that this maneuver reduces the storage space for $[N]$ to about half that required in Case 2. Row 4 shows that the translating table for part numbers occupies from 10 per cent to 20 per cent of the space required for $[N]$. It is to be noted that the storage space required for the translating table is not to be charged solely to the reduction in storage space for $[N]$. This single table effects reductions in storage space for all the matrices and schedules having one or more dimensions represented by part numbers.

The disadvantages of using an internal part number in addition to the external numbers are (1) storage space is required for a translating table, (2) all inputs and outputs connecting with people must be translated, and (3) the translation requires time and introduces the possibility of errors in translation. The errors in input translation will be detected by the input checking procedures described above. Similar checks on outputs might be instituted, checking control totals after the translation to external part numbers. The advantages of using short internal part numbers include (1) considerable net saving in storage space, (2) great savings in computation time, due to the fact that most of the processing can be carried on with the internal part numbers, using much shorter lengths of storage medium, and (3) easier comparison of part numbers during the body of the processing, especially if the machine is a fixed word length processor and the part numbers are greater than one word in length.

The storage space required to carry the checking column for N is:

$$M_{Nch} = Q_1(D_{Nch} + D_p). \tag{20}$$

Assigning 5 digits to D_{Nch} for example A, and 6 digits for examples B and C, we obtain the digit storage requirements for checking, given on the fifth row of Table IV. The space required varies from 7 per cent to 14 per cent of the storage space required for $[N]$.

TABLE IV
STORAGE REQUIRED FOR VARIOUS ORGANIZATIONS OF $[N]$

		Example A	Example B	Example C
Case 1	D	18×10^6	$5,000\times10^6$	$12,800\times10^6$
	C	$.027\times10^6$	$.7\times10^6$	1.1×10^6
	FT	20,000	5,600,000	14,000,000
Case 2	D	$.06\times10^6$	1.5×10^6	4.8×10^6
	C	$.30\times10^6$	11×10^6	24×10^6
	FT	410	14,000	31,000
Case 3	D	$.19\times10^6$	5.5×10^6	13×10^6
	C	—	—	—
	FT	210	6,100	14,700
Translate table	D	$.012\times10^6$	$.25\times10^6$	$.25\times10^6$
	C	$.027\times10^6$	$.70\times10^6$	1.1×10^6
	FT	40	1,100	1,500
Checking	D	$.027\times10^6$	$.55\times10^6$	$.88\times10^6$
	C	—	—	—
	FT	30	610	980

D = Digits, C = Characters, FT = Feet of magnetic tape.

The magnetic tape storage requirements are based on an assumed 100 characters per inch, with 75 per cent of the tape length containing information. The figures stated assume storage of data at the full density possible. In practice, considerations such as fixed word length, fixed block length, ease of programming, etc., will reduce the storage efficiency and increase the storage tape length required.

The major storage areas required for this problem are summarized in Table V. The figures given for matrices and schedules include check rows and columns. Here again, actual storage requirements will be considerably larger than the quantities quoted, for reasons such as those given at the end of the previous paragraph.

PROCESSOR CHARACTERISTICS

The above sections have defined the type of computation required and the volume of data involved in typical examples. This information makes it possible to state certain data-processor characteristics which are necessary or highly desirable in processors applied to the work described here. A general method of performing the data-processing task has been suggested. This method has been postulated keeping in mind the general characteristics of data-processing equipment now available or soon to be available. The method described does not, however, confine itself to the abilities of existing equipment; indeed, the method described is probably not wholly feasible on existing equipment, whose limitations will usually dictate departures from the direct method described here. Such departures will nearly always increase the time required to perform the computation. As a result of these considerations, a few desirable characteristics suggest themselves. These are given below. The fact that some of the desirable characteristics stated below are not available in existing equipment does not preclude the performance of this type of problem on available processors. The matters suggested below are given to point out characteristics which will sharply increase the speed with which problems of this type and size can be performed on automatic data-processing equipment.

TABLE V
STORAGE REQUIRED FOR THE MAJOR PARTS OF THE PROBLEM

	Millions of characters		
	Example A	Example B	Example C
$[N]$	.22	6.1	14
Part number translation	.039	.95	1.4
$[T]$	.15	.40	4.9
List of shippable articles A_k	.0013	.0047	.0038
s_k^m	.043	.056	.074
$[\sigma]$	.15	.38	3.9
x_i^m	.56	4.8	14
$[\tau]$	.087	2.8	8.0
Machine type translation	.004	.0009	.0009
h_n^m	.0043	.0057	.026
	1.26 1,400 *FT*	15.5 17,000 *FT*	46.5 52,000 *FT*

Type of Characters

The processor should be capable of processing alphanumeric characters, plus several other symbols encountered in part numbers (including dash, slash, and blank). This capability is necessary if the part "numbers" used internally are the same as those used by people. If the part numbers are translated into shorter numerical part numbers, as is strongly suggested above, it is still highly desirable to be able to handle nondigital characters within the machine, so that the processor can bear the burden of the attendant translating tasks.

Word Length

A variable word length machine can be used to some advantage here, providing a small advantage in storage efficiency compared with fixed word length storage, and providing a considerable reduction in programming maneuvers required to extract and shift parts of words.

If a fixed word length machine is to be used, we can get some idea of the desired word length by considering the storage organization suggested in the section on magnitude of the problem. In general, the data in the various matrices and schedules is stored with each element accompanied by one co-ordinate. On this basis, for nearly all of the storage required, the examples given here and the worst cases to be expected in practice require no more than 10 characters per word. One of the exceptions is the set of schedules s_k^m. Here we postulated the storage of accumulative shipping quantities along with the quantity per period and the period designation m. In two of the examples, this calls for 12 characters, and up to 15 characters are indicated for larger problems. Since this is but one part of the total storage problem, it should not be given great weight. For s_k^m, two adjacent words may be required for each element. Finally, the translating tables for part numbers, machine-type designations, and types of labor would often require very lengthy words for the numbers and designations used by people. There exist part numbers up to 25 characters in length. A widely encountered part number length is the 14-character Army-Navy number. Such lengthy numbers can be fitted into several shorter fixed-length words in the translating tables.

Fixed or Floating Point

For the purposes of the computation described here, a fixed point machine is satisfactory. In any given factory, the range of magnitude of the elements of each matrix or schedule is well known, much of the work involves small integers, and the range of the intermediate and final results is well behaved.

Form and Size of Major Storage

The figures at the bottom of Table IV give very rough measures of the total storage required in this work. For the largest example shown, storage of the order of ten to fifty million characters is required. Of the forms of storage available at present, magnetic tape appears to be the only one suitable for problems of this magnitude, for reasons of both volume of storage medium and input-output time.

A very general idea of the number of tape reels which should be electrically connected to the processor can be gained by considering one part of the process, such as the development of the $[T]$ matrix. During this computation repeated runs are made through $[N]$ and $[T]$. There are as many passes through $[N]$ as there are levels in the Gozinto graph, or q_5 passes. There are as many passes through $[T]$ as there are articles A_i, or Q_1 passes. By the end of the computation of $[T]$, it occupies more storage space than $[N]$ by the ratio of q_2 to q_1. In addition, the reduced form of $[T]$, containing only columns for shippable articles of A_k, has been developed and written out. If these repeated passes through the tapes are interrupted by manually changing reels thousands of times, the efficiency of the operation will be seriously impaired. It is desirable that magnetic tape storage capable of handling $[N]$ and both versions of $[T]$ be available.

Assume a tape storage density of 100 cells per inch, with 75 per cent of the tape length actually containing blocks of information. Assume further that the digit storage requirements stated in Table IV are placed in words and blocks in such a way that an 80 per cent storage efficiency is attained. Under these conditions, the tape storage desired in these examples is: A, 100 feet; B, 20,000 feet; C, 55,000 feet. Taking into consideration the fact that separate sets of data should be on separate reels (i.e., $[N]$ and the two versions of $[T]$ should be separated), the number of 2,500-foot reels indicated is: A, 3; B, 10; C, 23.

The order of 11 to 20 hoppers of some kind (e.g., tape reels, tape loops) is desirable for ease and efficiency of sorting.

Internal Storage

A very rough programming of this work and time estimates based on the present state of the art indicate that an "immediate access" storage of the order of 6,000 ten-digit words is the minimum desirable for problems of the magnitude of examples B and C. This should be backed up by a cyclic memory of the order of 20,000 words.

Instructions

The extensive matrix multiplication type of computation encountered in this problem argues for a multiply instruction which accumulates the sum of a series of products.

The performance of this problem includes several table look-up operations. An example of this takes place during the computation of $[T]$, where complete rows of $T_{p,j}$ are developed during the computation, but only the elements falling in columns representating shippable articles are to be written out in row form. Thus, the part number j for each element $T_{p,j}$ will be compared with a list of part numbers of shippable articles. If a storage with a 5-millisecond average access time is used to store the table (e.g., a 6,000 rpm magnetic drum with one head per band) the table look-up time in the three examples would be: A, 25 minutes; B, 17 hours; C, 25 hours. This is one of the lesser table look-up tasks in this problem, and even the times quoted here would jump greatly if the number of shippable articles were increased by including large numbers of shippable spares. It is highly desirable to cut this time by two orders of magnitude, from the 5-millisecond average access to 50 microseconds. This suggests that storage space for such tables be of the "immediate access" type if possible. If the table is stored in such a memory or in a cyclic memory, a "table look-up" instruction or its equivalent is strongly indicated here. In cyclic memories, this instruction should be mechanized to compare the comparee with each word as it passes the reading station, starting at a specified address.

If a fixed word length machine is used, and matrix and schedule information is stored as specified here, a good part of the internal programming will be devoted to extracting a part of a word for comparison with another word or part of another word. This sort of thing will be performed continuously in determining the co-ordinate of each element as matrix multiplications are being performed, as certain columns of $[N]$ are sought in the computation of $[T]$, etc. This suggests the need for a comparison instruction which includes a means for stating which section of a word is to be compared. Similar arguments apply to other arithmetic and logical instructions performed in this work. Considerable time will be saved and programming details will be appreciably reduced if addresses in all instructions include a means for stating the beginning and end of any section of a word desired.

Output Printing

The schedules printed out are the production requirements schedules, x_i^m, the machine loading schedules, h_n^m, and the labor loading schedules, similar to h_n^m. Assuming a 120-column form with the identifying part number, machine-type number, or labor type number printed on the left, followed by the quantities in the schedules spread across the page with 3 columns separation between quantities, and using as many lines as necessary to state a complete schedule for each part number, etc., the following modest printing loads are estimated for the examples: A, 6,000 lines per week; B, 50,000 lines per month; C, 160,000 lines per month.

Appendix I

Verification of the Accuracy of Matrix Multiplication

Consider:

$$[A]\cdot[B] = [C]$$

Augment $[A]$ with an extra row along the bottom, composed of the sum of the elements of each column. These are $\sum_i a_{i,j}$. The dimensions of the resulting product $[A^*]\cdot[B]=[C^*]$, are

$$(i+1, j)(j, k) = (i+1, k).$$

So $[C^*]$ has an extra row along the bottom.

$$c_{i,k} = \sum_j a_{i,j}b_{j,k}$$

$$c_{i,k}^* = \sum_j a_{i,j}^*b_{j,k}.$$

The last row of $[C^*]$, $i=m+1$, is

$$c^*_{m+1,k} = \sum_j a^*_{m+1,j}b_{j,k}$$

$$= \sum_j \left[\sum_i (a_{i,j})b_{j,k} \right] = \sum_j \sum_i a_{i,j}b_{j,k}$$

$$= \sum_i \left[\sum_j a_{i,j} b_{j,k} \right]$$
$$= \sum_i c_{i,k}.$$

Thus, the elements of the last row of $[C^*]$ are the sums of the elements of the columns above the elements of $c^*_{m+1,k}$. By augmenting $[B]$ with an extra column on the right, composed of the sums of the elements of each row of $[B]$, an extra check column of $[C^*]$ can be generated. Now, an erroneous element in $[C]$ can be located by noting the row and column in which the sums do not check in $[C^*]$.

Example 1

Using a check row in $[A^*]$, producing a check row in $[C^*]$:

$$\begin{array}{cccc} 0 & 1 & 0 & 3 \\ 0 & 0 & 2 & 1 \\ 3 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 \\ \hline 4 & 3 & 3 & 5 \end{array} \cdot \begin{array}{cccc} 1 & 0 & 0 & 1 \\ 1 & 2 & 3 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ & & & \end{array} = \begin{array}{cccc} 4 & 5 & 6 & 3 \\ 1 & 1 & 3 & 1 \\ 4 & 1 & 1 & 4 \\ 3 & 4 & 7 & 1 \\ \hline 12 & 11 & 17 & 9 \end{array}$$

Example 2

Using a check row in $[A^*]$ and a check column in $[B^*]$, producing a check row and column in $[C^*]$:

$$\begin{array}{cccc} 0 & 1 & 0 & 3 \\ 0 & 0 & 2 & 1 \\ 3 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 \\ \hline 4 & 3 & 3 & 5 \end{array} \cdot \begin{array}{cccc|c} 1 & 0 & 0 & 1 & 2 \\ 1 & 2 & 3 & 0 & 6 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 4 \\ & & & & \end{array} = \begin{array}{cccc|c} 4 & 5 & 6 & 3 & 18 \\ 1 & 1 & 3 & 1 & 6 \\ 4 & 1 & 1 & 4 & 10 \\ 3 & 4 & 7 & 1 & 15 \\ \hline 12 & 11 & 17 & 9 & 49 \end{array}.$$

Application of Data Processors in Production

C. R. DeCARLO[†]

THE PAST year has seen the firm establishment of the large electronic computer in industry. With the delivery and use of eighteen IBM 700 series of Electronic Data Processing Machines, as well as the delivery of machines by other suppliers, the electronic data processor has found a large measure of acceptance among American business management.

These large computers were originally conceived, developed, and installed as scientific or engineering aids. Indeed they have been of invaluable assistance in the aircraft and oil industries, in government scientific laboratories, particularly those dealing in nuclear power and weapons, as well as playing a significant part in the solution of important logistical problems.

As the data processing and computing capabilities of these machines became known to an ever-increasing number of people outside the scientific field, it was natural that attempts should be made to utilize them in solving problems in production and accounting.

However, the large electronic data processor, with its high arithmetic speed, capacious storage, and high-speed reading and writing, poses not only many opportunities, but also some problems in its application to production and accounting functions. Questions arise concerning the precise definition of logic used in a particular application. The availability and validity of data necessary to a machine operation are not always guaranteed. Often present departmental responsibilities must, or should, be consolidated in order to extract maximum efficiency from the data processor. These and many other problems require wisdom, mature judgment, and much patience on the part of the management group first approaching electronic data processing.

Parallel with the development and use of the electronic computer, and its appreciation by production managers, there has been developing a body of knowledge concerning the fundamental nature and theory of the production process.

This has been brought about largely through the efforts of mathematicians and production experts attracted to theoretical considerations; perhaps by the existence of the large-scale computer—a device which can serve as a laboratory to test their theories, as well as a tool for the ultimate implementation of their work.

Already several interesting and important applications of large electronic data processors have been made to particular production problems. One of these occurs in the aircraft industry, where a customer is using the IBM Type 701 EDPM for production scheduling and the control of shop orders.

The origin of any problem in manufacturing control is the production forecast, sales forecast, or contract commitment. These tell what must be made and when. From this will follow the computation of gross requirements, net requirements after processing with inventory files, and capacity requirements after scheduling.

In the case of an airplane manufacturer, the procurement of a contract is the starting point. A master

[†] International Business Machines, New York, N. Y.

schedule is set up for the contract stating the number of ships, rates of production, and number of days a given rate is to be effective.

The Type 701 prepares a Base Schedule which shows, by shop calendar date, the exact day each ship is due at a given station or line position on the production line. This schedule may be revised as manufacturing schedules accelerate or retard. In this particular customer application, 4,000 schedule pages were required to describe all models.

Dependent upon the type of schedule, processing time by previous methods was from one to four weeks. Through the use of the Type 701, schedules are completed within a 24-hour period at a 40 per cent reduction in preparation cost. This minimizes the number of shop orders which must be located and changed because of schedule revision. Because the preparation is done automatically at high speed, it is possible to evaluate several schedules to study the effect certain types of changes might have on resulting detail schedules. This enables the selection of better schedules.

Following the computation of the base schedules, ships of a given contract are grouped into lots for manufacturing. A schedule is prepared showing the manufacturing day on which the first ship of each lot moves through each position. Thereafter all periodic fabrication and assembly parts are scheduled to the first ship of the lot.

The ultimate creation of a Shop Order, authorizing the production of a certain part, derives from the base schedule as a result of processing a large main tape file equivalent to the following types of files normally found in production work:

1. Engineering and manufacturing parts list,
2. Parts history file,
3. Spare parts requirements file,
4. Parts inventory file,
5. Contract and work order cost file.

A file is prepared weekly reflecting changes in the main parts and schedule file. These changes are due to factors common to most manufacturing processes. They include:

1. Parts requirements changes,
2. Engineering changes,
3. Manufacturing practice changes,
4. Quantitative or transaction changes, losses, unplanned receipts, disbursements, etc.

This weekly change file is processed with the main tape file and a completely new revised main tape file is written. Calculations are made during this processing. Engineering and manufacturing changes are entered in the main file. Part usage by assembly and plane is extended and summarized by lot and contracts. Net quantities are computed for each lot in each contract. Previous part orders are examined and if a need for a part is determined, it will be scheduled and written on an output tape, to later prepare a Shop Order.

A scheduling operation is performed, and the need for part orders on succeeding lots is determined. The machine will continue to schedule quantities for each lot until a lot is reached which does not require the present initiation of shop orders. Finally, orders are written for those parts which require replacement.

As a result of this processing, various other information is obtained. This includes an output tape which is used to print the part activity ledger.

Another output tape is used to print shop orders and also generate a prepunched Open Order Card which will be used to introduce the completion of the particular order at a subsequent stage in the main file. Other miscellaneous output, such as unmatched transactions, reduced requirements information, or removals, is available on a tape for listing and action.

It is interesting to note that this system of processing represents a use of the "synthesis" method of computing production requirements. This method is similar to the regular product of the parts assembly matrix by the production schedule vectors. It differs in that additional vectors, representative of common parts usage, are appended to the former and corresponding elements added to the latter.

However, perhaps the most important concept illustrated by this use of the Type 701 EDPM lies in the consolidation of many files, all basically related to the part number, into one file, and the weekly processing of this file against one change file representative of all types of change, to create a variety of output data and automatically revise the main file to current status. Thus engineering, manufacturing, inventory and planning information are all treated simultaneously, making for better management.

The magnitude of this application can be appreciated if we consider the number of parts (almost 100,000) and the number of weekly changes (approximately 45,000). The main file consists of approximately 47 million digits held on 27 tape reels. Changes amount to 6 million digits held on 5 tapes. At least 8 output tapes totaling 12 million digits will be prepared. The total processing time for the complete application is under 40 hours.

A recent and important application of the large electronic computer in production work was developed by the IBM Endicott, N. Y., Production Division in cooperation with the IBM Laboratory Computation Service in Endicott. Here the IBM Type 701 was used to derive the Predicted Machine Load and Raw Material Forecast for the Endicott plants for the year 1955.

Basic to this application is the fact that the inventory control system is predicated upon an Ordering Cycle System. Thus the present inventory position is adjusted for released and back orders and minimum stock level before determining the Re-order Points. Combining the Re-order Points with scheduling data and production forecasts, it is possible, by reverse scheduling techniques, to obtain the load and raw material requirements for the plant.

In actual practice, the inventory position of a part as of a particular manufacturing day is analyzed by the machine. Parts on order are added to the present stock, and from this sum the amount of back-order parts is subtracted. This quantity is divided by the average daily usage for the particular part to give the normal production time interval covered by the present inventory position. From this is subtracted a number of days equivalent to the protective stock or minimum stock level. This, then, is the actual production time interval before replenishment is required. This interval, expressed in days, is added to the current manufacturing calendar date to determine the first actual stock date; i.e., the date at which new parts should be available.

These operations are expressed in the simple algebraic equation

$$\frac{S+O-B}{R} - P + D_c = D_0, \tag{1}$$

where

S = current stock
O = on orders
B = back orders
R = average daily use
P = protective stock (expressed in days)
D_c = current manufacturing calendar day
D_0 = first stock date.

As a part of our manufacturing process, the concept of Economic Order Quantity is used. This involves mathematical determination of the lot size which minimizes the total variable cost associated with inventory charges and setup or procurement expenses. For each part, the economic lot size is divided by the average daily requirements to determine the replenishment interval for the part. This is expressed as

$$\frac{E}{R_d} = I, \tag{2}$$

where

E = economic lot size
I = replenishment interval.

At this point the initial stock date and replenishment interval for a particular part have been computed.

The engineering part history, or "part routing," is now introduced. This is equivalent to operational estimates for machine tools or conversion processes and labor and raw material. Typical routing operations for each part include setup and production rates, raw material per part, transit times, etc.

The 701 now examines the last operation used in the part routing. The hours required in this operation for the particular part are computed, e.g., multiplying economical lot size by hours per piece and adding setup time. These hours are rounded to the next highest day. The days spent in the operation are then subtracted from the first stock date for the part to determine the start date for the part in the operation.

The replenishment interval is now added to this start date to determine each start date for the part in the given operation for the total scheduling period. Thus, a series of start dates for a particular part through a particular operation is obtained. The 701 determines when a start date falls beyond the range of the terminal day of the schedule period, terminating the computation.

Returning to the first operation start date, the 701 subtracts transit time for the part and examines the next preceding operation. Load computations are made, the replenishment interval added, and the process is repeated until the initial operation for the part is reached.

Parallel with this operation, load totals for each operation are accumulated.

As raw material is normally required at the first operation in a production sequence, it is also possible to calculate the amount required by multiplying raw material per piece by the economic lot size. As a convenience, these quantities are distributed by raw material type at some time interval, such as quarterly or monthly. Similarly, load hours are also accumulated by a 10- or 20-day time interval.

This application involved 30,000 parts in a 280-day schedule. The number of operational machine groups in the production process is 450. Approximately 3,500 different raw materials were used, distributed in quarterly intervals.

Prior to the present procedure, Static Load Forecast reports were run. These were developed on punched card equipment, utilizing the CPC. However, these did not take into consideration finished-parts inventories or "work-in-process" inventories. The processing required in this instance consumed approximately 3,200 hours and three million cards.

Using the Type 701, approximately 600 hours of conventional punched card equipment and one million cards were required to prepare card documents for proper 701 input. This will be unnecessary for subsequent reports, since the same documents prepared will be used again. The processing time expended by the Type 701 was 41 hours.

The uses of the output reports in this application are manifold. The Endicott plant management can determine whether present facilities are capable of handling projected building and shipping schedules. Requirements for additional facilities can be analyzed. A similar statement can be made with respect to manpower requirements. Surplus facilities are also indicated. Questions concerning the desirability of subcontracting versus facility acquisition are better answered as a result of this application. Finally, raw material ordering is considerably aided by the use of these reports.

Having now developed the machine program for this application, it will be possible to obtain more frequent load forecasts with a minimum of effort. Many management benefits are to be gained.

Prior to this application, an experiment was made using the 701 in New York for yearly schedules of manufacturing facilities. In this operation, the yearly requirements for a particular machine were to be scheduled, taking into account the capacities of various machine groups, and shifting the schedule where overtime occurred.

The production facilities comprised 40 machine groups which manufactured 700 parts. The production department knew the quantity of each part required for the annual production of machines. It had determined the number of times per year each part should be manufactured and the quantity that should be manufactured during each of the manufacturing periods. These values were based upon annual requirements and economic lot quantities derived from such factors as unit cost, cost of storage, size, ratio of setup to running time, etc. The conclusions were that the parts should be grouped into four classifications as to their frequency of production. Parts were to be manufactured monthly, quarterly, semi-annually, and annually.

Production lot quantities were determined so that when a production lot went into storage, only a 20-day supply remained from the previous lot. This reduction in inventory made it possible to store completed parts within assembly departments, thus eliminating the need for special storage facilities.

The actual objective of a good schedule, in this particular instance, was to make fairly uniform the loads on the machine tools used in the manufacturing of these parts. (It should be realized here that a schedule assuring uniform loading might not be a schedule with minimum cost.)

The 701 computed the schedule of each part through its proper sequence of operations and determined the actual day in the manufacturing calendar when a particular lot of parts should be at a work station.

An important output of the calculation was a set of IBM punched cards in which each card contained the part number, the machine group number, sequence number, and a breakdown of hours scheduled by days. The final printed report was a complete schedule of all parts showing start dates for each machine group.

Because the schedule existed in punched cards, it was possible to obtain two additional reports. The first, in which the schedule was grouped by machine groups, gave the machine floor manager an operating document describing the daily requirements of each machine group. The second, in which the schedule was grouped by start date, gave the stock analyst a document showing him when to initiate production orders and the jobs that should be on the machine for each day. Furthermore, reports of machine loading were available at any stage of the computation.

This application, while experimental, pointed up the need for a scheduling in reverse from stock dates, rather than presetting replenishment dates. The application was limited in that it did not treat all machine groups.

The Dynamic Load and Raw Materials Forecast can be extended in many ways. The obvious extension is to eliminate the notion of a linear usage rate and replace the set of averages by discrete functions dependent upon time. This will gear the inventory planning more closely with production planning. A second possible extension consists of recording the available capacity by machine group at each stage of the processing. At the first instance of overload, in the reverse scheduling technique, the machine could search through the last group of parts scheduled through the particular operation and shift these parts one day early in the schedule, with the possible effect of eliminating the overload. Obviously, this will increase in-process inventories; but it could smooth the use of production facilities. It would also be possible to formulate the decision rules necessary for the computer to know when such shifting should take place, as well as when certain excess machine time should be made available through the use of overtime.

The most important aspect of the applications described is that because they exist in the form of machine programs describing logical and mathematical models, it is possible to vary the input data and to observe the effect these changes will have on the total system, or in certain parts of the total system.

In the development of scientific methods in production and inventory management, there are five general areas in which research work will prove fruitful:

1. The development of stochastic models,
2. The improvement of engineering standards techniques,
3. Better sales and production forecast methods,
4. Extension of the present linear programming type of solution to allow the simultaneous estimation of time schedules while considering cost and profit functions,
5. Statement and solution of fundamental problems in combinatorial analysis which are required if scheduling techniques are to become subject to mathematical formulation.

In the applications described, the solution by the computer was predicated upon standard operational estimates of the production function. However, in reality, such things as machine breakdown, material availability, rejection rate, etc., all influence the actual fulfillment of a planned schedule. Such chance or stochastic variables can be treated through the use of mathematical probability methods.

An experimental application was programmed on a Type 701 to determine the feasibility of scheduling a precision manufacturing operation in a shop. Jobs in the shop were rated on a priority basis based on due date and the amount of work yet to be done. A group of jobs was chosen and these jobs scheduled, taking into account the chance variables of machine breakdown, operator performance, rejection rates, manpower availability and machine capacities. Each of these variables was brought into the solution by the use of density functions and random-number-generation techniques. Therefore, while the variables were random, their impact could be statistically bounded. For each permutation the schedule was computed four times with an average evaluation cost. Following this, other groups were scheduled in similar fashion, finally selecting that set of schedules which assured minimum cost.

A complete history was kept for a 12-week period of the operation of the shop. Information concerning each job number, sequence, and amount of time in various facilities, etc., was key-punched. An historical study was made by scheduling the job on paper for this same period, assuming the same information was available in the shop at the beginning of each week. Assuming the same sets of orders and facilities, it was estimated that there would have been a significant increase in production for this particular period.

This is an important application which can be extended in many ways. One of the most important aspects of this problem is in determining the validity of the mathematical and probability model chosen to represent the production entity.

In several cases, computers such as the Card-Programmed Electronic Calculator have been used to simulate production and inventory systems, taking into account a variability of the production functions. In one industry, which maintains inventory at widely dispersed points and has long shipping lines, inventory fluctuations are apt to be severe, depending upon the interrelationship of production and shipping times. A probability model was established, and the computer was used to determine the effect of chance variation in shipping schedules upon inventory fluctuations under a given set of inventory control limits. The result of this study was to demonstrate that the variability associated with these operations could be expressed mathematically, and that control limits could be set which would ease the problem of inventory fluctuations.

The second area of research concerns the field of engineering standards. Industrial engineering practices, in general, have assumed an averaging technique in introducing variability into operational estimates. More precise industrial engineering techniques will yield better standards for the use of production scheduling. Indeed, there has been very little statistical work done in the field of industrial engineering to express a standard properly. Obviously a proposed time schedule will be only as good as the engineering standards contained in the operational estimates.

Similar statements could be made with regard to the production forecast which gives rise to the production schedule. This area has received some attention in terms of market research and product usage studies.

The three areas mentioned above are concerned with statistical and probability methods. This field of mathematics offers great promise in industrial use. Another very important tool in mathematical programming is linear programming. This mathematical method has been used primarily in the solution of allocation problems where there are available certain resources and facilities capable of manufacturing a class of products. The mathematical problem here is to allocate the facilities and resources subject to linear constraints in such a way as to maximize or minimize a linear function of the resources and facilities so committed.

The Type 701 electronic computer has been used to solve problems involving the proper blending of gasolines, proper allocations of crudes to achieve maximum profit, and so on. There is also a class of problems involving the allocation of shipments in a transportation network. This problem has the same mathematical framework as the blending problem. It is a problem of extreme importance in any large, far-flung industry.

The important point to realize in the use of linear programming or allocative techniques is that they pose the problem as independent of time. What is needed is a method which will permit the simultaneous estimation of time-scheduled activities and cost or profit functions. The usual system of restraints must be capable of expression as functions of time: for example, expression of seasonal inventory policies.

The applications mentioned above, in which a computer is used to simulate the time-scheduling of production, are basically the framework upon which the mathematical programming must be expressed. At this time, some of the ideas contained in the theory of dynamic programming give promise of a start toward the solution of these problems.

The final area awaiting solution concerns the combinatorial problems which occur in production time-scheduling. Some research is being carried on to investigate interrelationships between cost and load functions and the combinations of job-loading in a typical manufacturing operation. The mathematical expression of the load function in terms of all possible combinations of job arrangements consistent with precedence relationships and capacity restrictions is a problem of enormous complexity.

It is pertinent to this paper to realize that the existence of the electronic computer has made it feasible for the first time to attack such a large-scale combinatorial problem. It appears that in the past, because of the utter hopelessness of enumerating even 10 combinations, problems involving such enumeration were generally ignored or solved by rudimentary methods.

The electronic computer, which makes possible the rapid enumerating of functions which are combinatory, gives the mathematician the first real challenge in this branch of mathematics. An insignificant amount of knowledge in this field is available as compared with the classical fields of analysis. Almost certainly, important research in production methods will develop in the near future as a result of the availability of the electronic computer.

The Integrated Use of Analog and Digital Computing Machines for Aircraft Dynamic Load Problems

B. MAZELSKY† AND R. F. O'CONNELL†

Summary—**This paper describes the techniques and methods used in investigating aircraft dynamic load problems such as vibration, flutter, gust loads, taxi and landing loads, based on an integration of both analog and digital facilities to provide maximum information on a problem in a minimum of time with a guarantee that the results are correct. The last requirement is an important consideration since high-speed computing machines not only provide the possibility of investigating far more complex dynamic problems but, unfortunately, also extend the probability of introducing errors due to the additional complexities of the problem.**

The advantages and disadvantages of both the digital and analog computers in terms of the needs specified for the aircraft dynamic problems are discussed; a method of analysis is suggested using the best features of each in a closely integrated manner. An example is presented of a typical dynamics problem for a transport type airframe.

INTRODUCTION

AS A RESULT of higher airplane speeds due to more powerful engines, and increased flexibility due to thinner lifting surfaces necessary for optimum performance, the aircraft designer must give considerably more attention to aircraft dynamics problems such as flutter, gust, landing and taxi loads in airplane designs. In many cases the representation of the elastic structure for adequate predictions of the vibration, flutter, and dynamic load characteristics of the airplane must be considered to a higher degree of complexity. Prior to the final acceptance of an analysis requiring the increased complexity, two important problems must be considered:

1. The elapsed time involved in determining a network of solutions, which will be shown to be a definite characteristic of dynamics problems due to required variations of the structure and the aerodynamics, should be well within the time the airplane design is "frozen" in order that the results obtained may be of use.

2. The results should enjoy some reasonable guarantee that they are correct, as well as make "engineering sense."

The realization of treating problems of a high degree of complexity, prior to the development of high-speed computing machines—both digital and analog, appeared remote. However, in recent years the computing facilities necessary to overcome these difficulties have not only been developed but apparently have been incorporated in the engineering forces of major aircraft companies. The purpose of the present paper is to review the possible approaches to the various phases of aircraft dynamics problems and also to examine the optimum methods in terms of both the analog and digital facilities for successful solution of these highly complex problems. The optimum method should be based on integrating both the analog and digital facilities such that the best features of each would be available to overcome the two previously mentioned problems.

† Lockheed Aircraft Corp., Burbank, Calif.

AIRCRAFT DYNAMIC LOAD PROBLEMS AND SOLUTIONS AT VARIOUS AIRPLANE DESIGN STAGES

As implied by the word "dynamic," the aircraft problems of interest in this paper are those which involve the element of time. Specifically they are aeroelastic problems such as flutter, airplane dynamic stability as affected by wing and tail elasticity, gust loads, as well as vibrations problems which involve landing and taxi load investigations. Examination of the characteristics of each problem indicates that a major portion of each is common to all of the problems. As a result, the study of a variety of phases of airplane dynamics could be made with the same computer setup with relatively minor modifications. The study of these problems can be broken up into three airplane design stages, as follows:

1. Preliminary design.
2. Shake test.
3. Final analysis.

Preliminary Design Stage

The preliminary design stage of an airplane offers the dynamicist the opportunity of determining the adequacy of a design which is primarily set by performance, stability, and maneuverability criteria at a time when the design is still not frozen. In the past the role played by the dynamicist at this design stage has been to offer an opinion, on the basis of experience on previous designs, as to the possible difficulties the configuration would encounter. These difficulties, if still present after the configuration is built, could be counteracted by beefing up the structure or by judicious use of balance or ballast weights. On the present and future designs such a philosophy may not be adequate and could result in a final configuration that is unduly penalized by an unnecessary weight increase, since an easy "fix" may not be possible.

The principal difficulty in determining significant engineering results in the preliminary design stage is the restricted availability and accuracy of the data, both structural and perhaps aerodynamic, if the design is radically different, requiring experimental testing to help define the aerodynamic coefficients. Two possibilities exist at this point:

1. The analyst can direct his efforts principally to refine the data to a higher degree of accuracy and therefore postpone the preliminary analysis until such time as the data is accurate enough to warrant an analysis that would lead to significant engineering results.

2. An analysis can be made based on available data realizing its inadequacies and shortcomings with the intention of first getting a "feel" for the problem and of determining what structural or aerodynamic parameters have the most effect on the solutions. In contrast, those parameters which have a relatively minor effect can also be determined. At this point further investigation of those parameters which must be known to a higher degree of accuracy for adequate engineering results can be pursued by more detailed study or by experimental techniques involving carefully scaled models.

In view of the limited time available for performing a successful analysis while the airplane is in the preliminary design stage, the second choice appears to be the most logical. During the investigation of the preliminary design an unsatisfactory condition in terms of stability, such as flutter, may appear. Through the use of many parameter variations the most logical fix, that is, the one requiring the least weight increase and complexity, can be determined. Recommendations could then be given to the Design Group for possible incorporation in the design. Thus an analysis in this stage has many desirable features. However, the following requirements must be dealt with if the results of the analysis are to be useful:

1. Large number of degrees of freedom.
2. Large number of solutions.
3. Limited time for determining solutions.
4. Guarantee that the solutions are correct.

Shake Test Stage

The next stage of importance to the dynamicist involves the vibration or shake test of the airplane. As many dynamicists know, the results obtained in these tests play an important role in the studies that follow: consequently, careful attention should be given to these tests to insure adequate correlation with the calculations made in the Preliminary Design Stage. Correlation of the observed and calculated vibration results allows for a means of determining the validity of the mechanical representation used. Differences that do occur can be attributed to two primary reasons:

1. Selection of proper as well as adequate number of degrees of freedom.
2. Accuracy of structural data.

Detailed comparisons of the modes are usually sufficient for evaluation of the first item. The second item, however, is not readily evaluated by mere inspection but usually involves numerous calculations involving a large number of variations of structural parameters. Such a network of solutions supplies the dynamicist with a means of determining the optimum match of the observed shake test results requiring the most reasonable adjustment of the initial structural data. The optimization should be based on not only the match of frequencies but also mode shapes, node lines, and relative phasings and amplitudes among the modes. Of course, this stage also presents a crisis (except in those fortunate cases where it is obvious that no serious dynamic problems exist) in view of the allowable time required to match the observed shake tests, since the first flight date of the airplane is not far off. A final analysis check has to be made prior to the first flight date and time must be allowed for this investigation. Thus the success of refining the analysis at this stage depends again upon the four previously mentioned requirements.

Final Analysis Stage

The final analysis stage permits the dynamicist to determine the effects that the changes in structural representation necessary to match the shake tests have on the final results. In addition a reorganization of the pertinent degrees of freedom may appear warranted as a result of the knowledge obtained in matching the observed shake test results together with the aeroelastic stability or dynamic load results obtained in the preliminary stage. For instance, the flutter investigations may indicate that the critical flutter speed of the airplane involves the coupling of essentially two or three vibration modes on a particular surface. The modes on the remaining surfaces or structure may have a relatively minor effect on the critical mode. Consequently, a more refined definition of the critical modes could be made by redistributing the degrees of freedom. In addition, the pertinent degrees of freedom for gust, landing, and taxi loads may not necessarily be compatible for optimum representation for each problem. Thus a certain amount of further reorganization (much of it should be already planned from the results of the preliminary analysis) may be necessary and must, of course, be done rapidly and accurately. Again the success of this stage depends upon the four previously mentioned requirements.

Analog and Digital Methods of Analysis

In describing a complex physical structure for the purpose of analysis by means of any computing technique, a primary consideration is that of reducing the given physical system to a system having a finite number of degrees of freedom. In general, all computational methods are restricted to the consideration of a limited number of degrees of freedom. The problem then becomes one of retaining all of the significant characteristics of the physical system, within the limitations imposed by the particular method employed. Various representations have therefore been evolved which accomplish this reduction. Since each of these representations is necessarily an approximation, the final evaluation of each must be made in terms of the particular application. Certain conclusions, however, may be drawn from a consideration of the general characteristics of some of these methods.

Analog Methods

Modal: A common means of representation of a physical system is through the use of a modal type analysis. Here the motion of the physical system is assumed to be represented by the linear combination of a limited number of modes. These modes must be chosen to include all the relevant types of motion of all co-ordinates of significant importance in the system. In practice, the modes chosen are usually the lower-order vibration modes (either normal or uncoupled) in the important degrees of freedom of the system.

The description of a mode of a distributed physical system involves the assumption of a deflection curve which is a function of a space-wise dimension. The deflection curve should approximate the true deflection of the system in the particular mode, and is chosen either on the basis of analytical or experimental results or from previous experience. Once this deflection curve has been prescribed, the generalized inertial, elastic, and aerodynamic properties of the system may be determined for motions of this mode. The determination of these generalized properties involves an integration with respect to a space-wise dimension. The differential equations of the modal system are then derived from the Lagrangian energy principles as applied to these generalized properties. Electronic differential analyzers such as BEAC or REEVES computers are normally used to simulate the resulting differential equations.

It will be noted that the modal approach results in a two-fold simplification of the distributed physical system:

1. The number of degrees of freedom is restricted to those of importance to the particular analysis.
2. The three-dimensional properties of the distributed physical system are replaced by equivalent two-dimensional properties.

Finite-Difference: As an alternative analog method, a distributed physical system may be represented by assuming the properties of the system to be concentrated at discrete intervals. This lumped-parameter system is then described by means of finite-difference equations, which are in turn represented by an analog computer. The computer which most commonly utilizes this method of representation is the network analyzer, or direct analog computer. The network analyzer utilizes passive electrical circuit elements to represent the discrete structural parameters of the approximate physical system. In the direct analog computer, both the electrical quantities, voltage and current, are analogous to quantities of the mechanical system. Two analogies are widely used—the loop analogy, wherein electrical voltage is analogous to mechanical force; and the nodal analogy, wherein electrical current is analogous to mechanical force. The nodal analogy is generally preferred in the representation of structural systems; this is due in part to the close topological similarity between the resulting electrical circuit and the structural system. Since mechanical quantities are usually translated directly into electrical quantities without reference to the differential equations of either the mechanical or the electrical system, this topological similarity is an important characteristic of the nodal analogy. The external forces applied to the mechanical system, e.g., aerodynamic forces, are represented by the analogs of the differential equations of these forces, much as is done in the case of the differential analyzer. In this case, however, current generators are required as well as voltage amplifiers, since current and voltage both represent mechanial quantities.

Although each of the two methods, modal and finite-difference, has been identified with a particular type of analog computer, variations of these methods are used in conjunction with either type. Differential analyzers have frequently been used to represent the finite-difference equations of distributed mechanical systems. More recently, a method has been devised wherein mechanical properties are simulated in differential form by elements of the differential analyzer, and these elements are interconnected in accordance with finite-difference equations of the mechanical system. In the case of the network analyzer, frequent use is made of modal representation in order to achieve a saving of electrical circuit elements. This is usually done in representing a subsidiary portion of the structure, such as a fuselage, where detailed information is not a required part of the solution.

Digital Methods

The methods used in the solution of aircraft dynamics problems with high-speed digital computers parallel quite closely those described in conjunction with analog computers. Modal type analyses are commonly used with digital computers in much the same manner as with analog computers, with one principal difference—the generalized co-ordinates in the digital analysis are usually assumed to be prescribed functions of time, i.e., to have only simple harmonic motions. In some cases, as for landing or taxi loads, limited time histories are computed for discrete forcing functions. The assumption of harmonic motion is certainly true in the case of the free vibrations of a conservative system, and while it is not in general true of a mechanical system under the influence of aerodynamic forces, the assumption is valid for those critical cases where the aeroelastic system becomes neutrally stable. In other cases, harmonic motions are obtained by the addition of an appropriate damping or driving force to the mechanical system. The differential equations of the modal system are then expressed as homogeneous equations containing an unknown frequency. The solutions of the characteristic equation of the system are then obtained by numerous methods such as matrix iteration, or fraction series, etc., and the frequencies so obtained are the desired vibration frequencies or flutter modes, as the case may be. The magnitude and sign of the required damping (if any) is then taken as a measure of the stability of the given root of the system.

Finite-difference techniques may also be applied to the solution of dynamics problems on digital computers. The mechanical and aerodynamic properties of the system are described in terms of discrete stations or panels of the distributed structure. The equations of motion of each station are expressed in matrix form in terms of the summation of forces at each station. The summation of these forces—elastic, inertial and aerodynamic—is performed by the use of integrating matrices which prescribe the distribution of these forces. This formulation results in equations of motion of finite-difference form which replace the partial differential equations of the distributed physical system. The equations obtained in this manner are time-dependent functions. As such, they are not in an expedient form for solution by digital means, especially when aerodynamic forces must be considered. The motions of the system are, therefore, normally restricted to harmonic oscillations and the equations written as functions of frequency. This set of equations is then solved as an eigenvalue problem by means of matrix iteration.

Comparison and Evaluation of Methods

Analog: 1. Modal Systems—The use of modal systems with analog computers has the advantage that it requires a minimum of electrical circuit parameters for a given complexity of structure. This feature is, indeed, the principal attraction of modal analyses. As might be expected, however, this simplicity of representation gives rise to other characteristics which are less desirable:

(a) The fact that the generalized structural parameters are obtained from space-wise integrations of the original system parameters means that localized structural or inertial changes are not simple to perform.

(b) The deflection curves which are assumed in modal analyses represent constraints on the system, which may have a significant effect unless the advantage of simplicity is lost by using many modes. The inclusion of these constraints assumes that external forces have no significant effect on the mode shapes.

2. Direct Analog Systems—Direct analog representations as applied to the network analyzer offer several formidable advantages:

(a) Complex structural systems may usually be analyzed without specific reference to the differential equations of the systems when the modal analogy is used, as is customary.

(b) Since the mechanical system is represented directly on the computer, the complete aircraft dynamics problem may be solved with one general setup. The changes in setup from static deflection tests to free vibrations, to flutter analysis, to gust response studies, etc., are accomplished in a very short time.

(c) The high degree of identification between electrical quantities results in extreme convenience in making localized changes of structural parameters. To expedite these parameter changes, the passive electrical circuit components of the network analyzer are variable over a wide range of values in small increments.

(d) The short time required for the actual machine solution, coupled with the ease of making parameter changes, makes possible the investigation of a wide range of structural configurations in a relatively short time. In this way, the significant parameters in a particular analysis are readily ascertained and investigated in detail.

The use of the network analyzer in the solution of dynamics problems is not, however, free of disadvantages:

The analysis of complex structural systems necessarily results in complex electrical circuits. Each element of the circuit representing the structure must be set to the appropriate value and correctly connected into the structure. The aerodynamic forces are simulated through the use of active elements, and these electronic devices are, of course, subject to failure or malfunction. One of the most difficult phases of the analysis is that of ascertaining that the setup is correct.

Although the electrical elements used in the network analyzer are the best quality obtainable, they are not perfect. The parasitic effects associated with these imperfections may be minimized by the proper choice of element values, but in some cases these effects are difficult to evaluate. This is particularly true of the small amount of damping which is necessarily present in any circuit containing inductors or transformers.

Since the network analyzer has been used principally in analyses employing classic two-dimensional incompressible theory, the use of other aerodynamic theories generally requires some development of the associated circuits. Although this is not an inherent limitation of the computer, it may be a serious difficulty if close cooperation between the aerodynamics group and the computer group is not maintained.

Digital: 1. Modal Systems—The use of digital computers in modal analyses is characterized by the same general advantages and disadvantages as were mentioned in conjunction with modal analyses with analog computers. The principal differences between these two methods are:

(a) The accuracy of solution is inherently better in the case of the digital computer.

(b) The digital computer requires a solution time which is considerably longer than that of the analog computer (that is, when many parameter variations must be considered).

2. Finite-Difference Methods—The use of finite-difference techniques with high-speed digital computers is in many respects similar to the use of direct analog representation. This method of formulation eliminates the necessity of prescribing deflection shapes; the motions which are represented are not limited by artificial constraint except due to the number of panels which may be included in the analysis. Successive stages of the

analysis—static deflection tests, vibration test, flutter—may be performed using digital programs which are very similar to those required for the analog. The resulting eigenvalue problem is solved by means of matrix iteration; this technique provides the means of discontinuing the solution after a specified number of roots are obtained. This number, of course, depends on the problem and its counterpart solution obtained on the analog for comparisons.

The time required for the formulation and setup of the problem is comparable to the time required for the same operation using the network analyzer. The principal advantage of the digital method over the corresponding analog method is due to the inherent accuracy of the digital computer. In general, the digital computer may be considered to solve exactly the problem which is programmed, while the network analyzer involves certain inaccuracies even under optimum conditions. The digital method, however, requires an appreciably longer solution time than the analog for repetitive solutions of the kind required by parameter studies. Although this longer solution time may not be significant if the number of solutions is small, the nature of the dynamics problem indicates that the number of solutions required is large. Therefore, the use of the digital method by itself is seriously restricted.

Suggested Method Based on Integration of Analog and Digital Machines

As noted previously, the analysis of a complex dynamic system usually requires a large number of solutions involving wide variations of the significant parameters of the system. These solutions, however, must often be obtained with a minimum of elapsed time, and must be free of both system errors and errors in machine setup. The finite-difference or "panel" method using the digital and analog computers in an integrated manner provides a means of achieving these desired results. The advantages of both computational systems are obtained —the ease of parameter variations and speed of solution using the direct analog computer as well as a continued check on the accuracy of the solution using the digital computer. The very fact that the analysis is performed by two independent methods is of course advantageous in assuring elimination of errors. The realization of the full capabilities of the integrated digital-analog method is most nearly attained when both computers use the finite-difference or "panel" approach. This formulation results in optimum compatibility between the digital and analog systems, and provides for comparative checks of the two systems at each stage of the analysis—elasticity, vibration, and aeroelastic stability checks.

Example of Integrated Analog and Digital Analyses

The integrated approach suggested in the preceding section was recently applied to the tail symmetric flutter analysis of a transport type airplane being built at Lockheed. The full capacity of the network analyzer was utilized as well as the maximum programming techniques available at the time of investigation on the IBM 701 computer. A sketch of the degrees of freedom is shown in Fig. 1. Six spanwise stations for the stabilizer

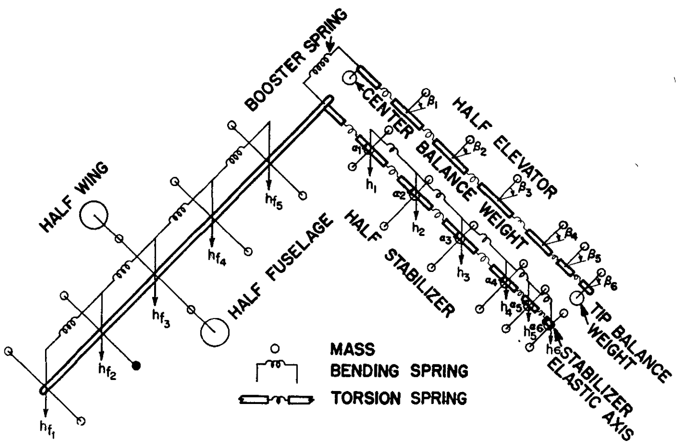


Fig. 1—Schematic of degrees of freedom for symmetric tail flutter.

and elevator combination were chosen as well as five sections for the fuselage. This combination provided six bending and six torsion modes on the stabilizer, six elevator flapping and torsion modes, as well as five fuselage vertical bending modes. Simple beam theory was used for representing the structure in bending and torsion. The elevator was assumed to have a flapping mode with a restraint determined by the boost system. The torsional rigidity of the torque tube required to transfer the elevator hinge moments into the fuselage through the boost system was also represented. In addition an elevator balance weight located at the fuselage centerline was assumed as a separate degree of freedom because of its large concentrated weight on the relatively flexible torque tube extending from the inboard end of the elevator to the balance weight linkage. The whole airplane was free to pitch and plunge. Unfortunately, the wing flexibility was not included due to the limited capacity of the computer. This problem illustrates very well the degree of complexity of airplane structure that can be handled by such a computing facility.

The first stage in the analysis was to check the stiffness circuits on the analog as well as the stiffness values used in both the analog and digital setups. This was accomplished by applying loads, moments, and hinge moments and recording the deflections. A comparison of the deflections was then made with those calculated from the digital finite-difference representations for the same loading conditions. These comparisons are shown in Fig. 2 for bending and torsion of the stabilizer as well as the elevator deflection. Note the remarkable agreement between the two approaches. No detailed struc-

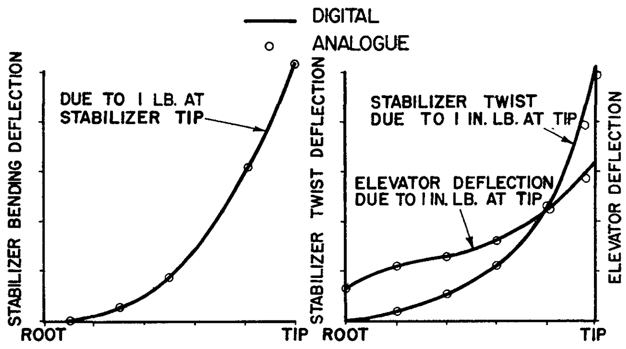


Fig. 2—Comparison of analog and digital static deflections due to unit tip load and moments.

tural deflection measurements were made for the fuselage. The elasticity checks, which were made concurrently on the digital and analog computers, took approximately one week from the time the structural data were made available to the engineers.

The next step in the integrated approach required the addition of the appropriate masses, inertias, together with the proper center-of-gravity locations of the masses in order that a vibration test correlation could be made. Since the fuselage modes were matched on the analog with observed shake tests, and no aerodynamics would be present on the fuselage, the correlations were made with the fuselage rigid and restrained in pitch and plunge. A summary of the natural frequencies obtained by the two computers is given in the following table.

TABLE I

Comparison of Analog and Digital Natural Frequencies

Analog (cps)	Mode	Digital (cps)
6.4	Stabilizer first bending	6.12
7.34	Elev. flapping (boost on)	7.30
16.8	Elevator first torsion	16.23
22.85	Elevator second torsion	23.1
28.0	Stabilizer second bending	26.4
32.1	Stabilizer first torsion	33.3

Note the close agreement between the two completely independent solutions. The differences in frequencies between the digital and analog are most likely due to the differences in finite-difference approximations used in the digital and analog representations. The analog representation implies trapezoidal type approximations while the digital consistently employed parabolic type approximations similar to Simpson's rule. It has been found that for the digital representation the use of higher order finite-difference approximations does not complicate the computations; for the analog, more refined finite-difference approximations, while possible, are usually not practical due to the excessive use of electrical components.

The corresponding mode shapes and relative phasing were also compared. A typical comparison is shown in Fig. 3 for the stabilizer mode. Examination of the results in Fig. 3 as well as those for the other modes indicated that, in general, the mode shapes and relative phasings agree remarkably well; the relative amplitudes, however, experience appreciable differences. These differences are primarily a result of the inherent "structural" damping of the order of one to two per cent present in the analog computer. The corresponding digital calculations were made, of course, with no structural damping.

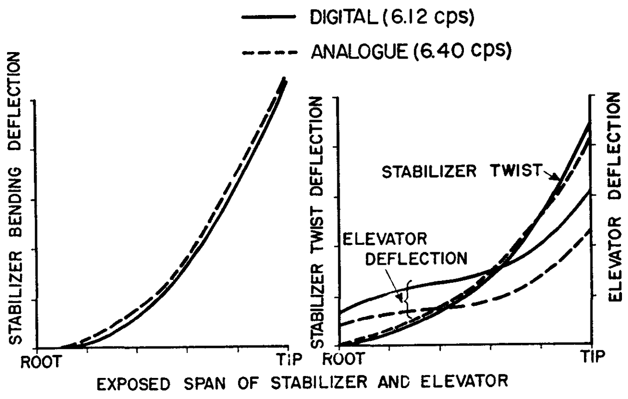


Fig. 3—Comparison of vibration modes for stabilizer first bending determined on digital and analog computers.

An estimate of the time involved to set up this problem on the analog and digital computer was made. Since both setups are made completely independent of each other, both problems were solved simultaneously and took approximately one week from the time the elasticity correlations were made. Parameter variations were made on the analog for several days to determine the adequacy of the degrees of freedom selected as well as the structural data determined from preliminary calculations.

Once the elasticity and vibration correlations were made, the transient aerodynamics were applied to the stabilizer and elevator to determine the flutter characteristics. The significant results are plotted in Fig. 4 as

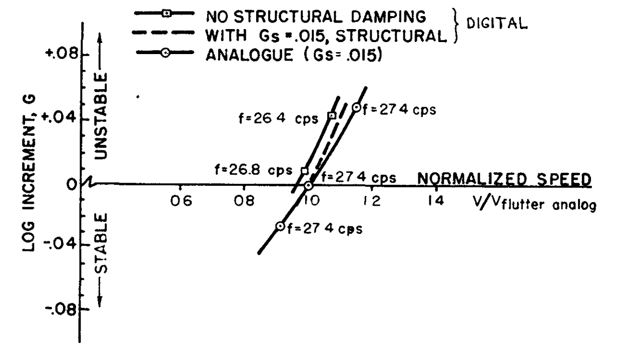


Fig. 4—Comparison of stability diagrams (*V-G*) determined by analog and digital computers.

a function of logarithmic increment of damping versus speed, normalized to the flutter speed obtained on the analog. Although other modes were present on both the analog and digital solutions, the one shown in this figure represents the most critical; that is, the mode having the lowest flutter speed. Also shown are the frequencies of oscillation for the analog and IBM at various stability and speed conditions. Comparison of the two solutions indicates that the frequency of oscillation at flutter agrees remarkably well. The flutter speed for the analog, however, is higher than the speed obtained by the digital setup. This difference can be explained by adjusting the digital calculations to correspond to the amount of inherent "structural" damping present in the analog. An estimate of the analog damping is approximately $g_s = 0.015$. The original digital calculations correspond, of course, to no structural damping. If a value of damping, $g_s = 0.015$, is used for the digital calculations, the revised digital stability curve corresponds almost identically to the analog curve. Thus the necessary correlation has been established. At this point the effect of numerous variations of structural or aerodynamic parameters on the critical flutter speed can be investigated in a relatively short time.

Approximately one to two weeks is required to obtain both the analog and digital correlations.

Concluding Remarks

This paper attempts to define the major phases of dynamic loads problems faced by the dynamicist at various design stages of the airplane. At each stage the nature of these problems was investigated and the following characteristics were found to exist for each:

1. The dynamic problems are becoming increasingly complex due to higher airplane speeds and increased flexibility due to thinner lifting surfaces necessary for optimum performance.
2. The amount of time required for obtaining accurate results which make good "engineering sense" is critically limited since accurate structural and aerodynamic data is not readily available.
3. As a result of the lack of data a large number of parameter variations are required in the preliminary design stage. Examination of the other two stages—shake test and final analysis—also indicates the requirement of a large number of parameter variations.
4. As a result of the increased complexity of the dynamics problem the results must be reasonably guaranteed to be free of errors.

In view of these characteristics, a review of possible methods using high-speed computing machines—both digital and analog—was made and an approach was determined using both the analog and digital computing machines in an integrated manner. An example illustrating the use of both types of computers is given, indicating a means of rapidly correlating the results obtained as well as permitting the dynamicist to use the best features of each computer to meet the stringent requirements necessary for a successful analysis.

A General Digital Computer Program for Static Stress Analysis

P. H. DENKE AND I. V. BOLDT†

Summary—**In a digital computer installation devoted primarily to the solution of aircraft design problems, an attempt has been made to provide the engineer, whenever possible, with programs which are general enough to be applicable to a large class of the problems which he typically encounters. Such a general program has been devised for the stress and deformation analysis of aircraft structures. The mathematical formulation of the problem and the program for its solution are discussed, and some selected applications of the analysis to actual structures are presented.**

Introduction

THE EXISTENCE of the high performance digital computer and the introduction of matrix algebra into structural analysis have created conditions favoring the development of a general program for solving static stress problems. To be most useful such a program should be applicable to any structure without additional programming time, should require as input data no more than the basic geometric and elastic parameters which define the structure, should be expressed in terms familiar to the practicing stress analyst, should require a minimum of machine time with provision for rapidly taking into account changes of stiffness and loading, and should permit the accurate analysis of highly complex indeterminate structures. The following paper describes a program which was designed with these requirements in mind.

† Douglas Aircraft Co., Santa Monica, Calif.

Static Stress Analysis

The problem considered is the stress and deflection analysis of statically indeterminate structures. When either the Maxwell-Mohr or the least work approach is adopted, the solution of the problem usually requires the following steps: (1) Given the co-ordinates of the structure, compute certain geometric constants, such as direction cosines and moment arms of forces; (2) write the equations of equilibrium; (3) write the equations of continuity. The solution of the equations of equilibrium and continuity completes the analysis of the structure.

The equations appropriate to the analysis of indeterminate structures have been known for many years and have appeared in a variety of forms. However, the introduction of digital computing machinery has led to new developments, one of which is the use of matric notation. The matric formulation of the problem is rather recent and has occurred historically in what seems like reverse order; that is, the equations of continuity have come first. The matric continuity equations have appeared in the literature in several places.[1–3] Denke has given matric formulations of the Maxwell-Mohr equations which are general enough to account for the effects of thermal deformation and certain nonlinear effects.[4]

However, the matric formulation of the continuity conditions still leaves a large portion of the problem unsystematized and poorly adapted to machine computation. This fact was pointed out by Langefors,[5] who developed a method of generating the matrices required in the continuity equations for beamlike structures from matrices established for single cells. This matter was given further study by Denke,[4] who developed in very general terms the matric equations of equilibrium and certain linking equations which permit generation of the continuity matrices from the conditions of statics. This work virtually completes the matric formulation of the equations of equilibrium and continuity. There remains the geometric problem mentioned previously. In the following discussion the matric equations of continuity and equilibrium are reviewed, and a systematic method of generating the equilibrium matrices from the structural geometry is presented. These geometric equations allow machine computation to begin at an earlier stage of the solution, and greatly simplify the work of the analyst.

The Continuity Equations

Enough redundant constraints are removed (or "cut") to produce a statically determinate structure. Then in Denke's notation,[4] the internal forces and deflections of the indeterminate structure are given by

$$F = [I - f(f^TDf)^{-1}f^TD]F_0, \tag{1}$$

and

$$\Delta = f_\Delta{}^TD[I - f(f^TDf)^{-1}f^TD]F_0 \tag{2}$$

where

F = a matrix of forces in the indeterminate structure,

F_0 = a matrix of forces in the determinate structure resulting from the external loading,

f = a matrix of forces in the determinate structure resulting from unit values of the redundants,

f^Δ = a matrix of forces in the determinate structure resulting from unit dummy loads,

Δ = a matrix of deflections of the indeterminate structure, and

D = a matrix of member flexibilities.

The Equilibrium Equations

The statically determinate structure is broken into free bodies. Associated with each free body are a number of degrees of freedom, and acting upon the various free bodies are a number of force pairs and reactions. Denke[4] explains the usage of the terms "degrees of freedom" and "force pairs."

The matric equilibrium equations are

$$Q_0 = -M^{-1}P_0, \quad q_\Delta = - M^{-1}P_\Delta, \quad q_x = - M^{-1}P_x, \tag{3}$$

where

M = a matrix of the components in the various degrees of freedom of unit values of the force pairs and reactions,

Q_0, q_Δ, q_x = matrices of statically determinate force pairs resulting from external loads, unit dummy loads, and unit redundants, respectively, and

P_0, P_Δ, P_x = matrices of components in the various degrees of freedom of the external loads, unit dummy loads, and unit redundants, respectively.

Eqs. (3) give the statically determinate internal loads in terms of the external loads and the matrix of "generalized co-ordinates" M.

The Linking Equations

The continuity matrices are related to the equilibrium matrices by the equations

$$F_0 = NQ_0 + H_0, \quad f_\Delta = Nq_\Delta + H_\Delta, \quad f = Nq_x + H_x. \tag{4}$$

These transformations are required because the force pairs used in solving the equilibrium problem are not necessarily the same as the member "forces" required in the continuity analysis. Note that the "forces" referred to in (1) to (4) are generalized forces and represent either forces or moments.

The Geometric Equations

Eqs. (1) to (4) comprise a nearly complete matric formulation of the indeterminate structures problem. In order to start the analysis, however, one must have the matrices D, N, H_0, H_Δ, H_x, M, P_0, P_Δ, and P_x. The matrix D contains the element flexibilities and is computed from basic data. The computation of the matrices

[1] B. Langefors, "Analysis of elastic structures by matrix transformation with special regard to semi-monocoque structures," *Jour. Aeronaut. Sci.*, vol. 19, pp. 451–458; July, 1952.

[2] L. B. Wehle and W. Lansing, "A method for reducing the analysis of complex redundant structures to a routine procedure," *Jour. Aeronaut. Sci.*, vol. 19, pp. 677–684; October, 1952.

[3] H. Falkenheimer, "Systematic analysis of redundant elastic structures by means of matrix calculus," *Jour. Aeronaut. Sci.*, vol. 20, p. 293; April, 1953.

[4] P. H. Denke, "A matric method of structural analysis," Proc. Second U. S. National Congress of Applied Mechanics, 1955.

[5] B. Langefors, "Matrix methods for redundant structures," *Jour. Aeronaut. Sci.*, vol. 20, p. 292; April, 1953.

N, H_0, H_Δ, and H_x is simple and often trivial. The matrices M, P_0, P_Δ, and P_x can be computed by the method outlined in the following paragraphs.

At this stage it is convenient to introduce a second set of internal forces called "subordinate force pairs" related to the principal force pairs by simple transformations. It is necessary to make such a transformation when, for example, it is desired to utilize the notion of shear flow in a panel (the principal force pair) in place of the four forces acting on the edges (the subordinate force pairs). Similarly, a single external loading condition (corresponding to a column of P_0) can be thought of as a principal generalized force, while the individual external loads can be regarded as subordinate forces. Similar remarks apply to deflection forces and redundants. Subordinate forces and quantities associated with them are indicated by a prime (′) attached to the symbol.

By definition, then, the matric element m_{ij}' represents the component in the ith degree of freedom of a unit value of the jth subordinate force pair. These "generalized co-ordinates" may be classified as "translation-force," "rotation-moment," "rotation-force," or "translation-moment," depending on the type of degree of freedom and force pair. Fig. 1 shows a reference frame,

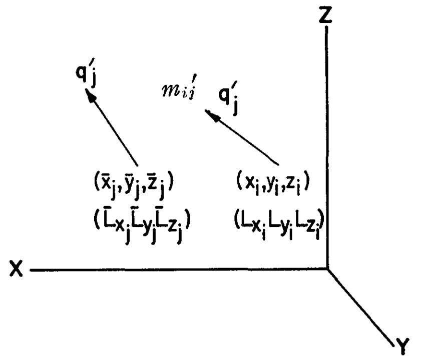

Z
q_j'
m_{ij}'
q_j'
$(\bar{x}_j, \bar{y}_j, \bar{z}_j)$
$(\bar{L}_{x_j} \bar{L}_{y_j} \bar{L}_{z_j})$
(x_i, y_i, z_i)
$(L_{x_i} L_{y_i} L_{z_i})$
X
Y

Fig. 1—Force pairs and degrees of freedom.

one member of a subordinate force pair, and the component of that force pair in a given degree of freedom. The figure shows the notation for co-ordinates and direction numbers. The co-ordinates are the co-ordinates of arbitrarily chosen points on the lines of action of the force pair and degree of freedom. The generalized co-ordinates are given as follows for the four cases.

1. Translation-Force

$$m_{ij}' = C_{ij}(l_{x_i}\bar{l}_{x_j} + l_{y_i}\bar{l}_{y_j} + l_{z_i}\bar{l}_{z_j}). \qquad (5)$$

2. Rotation-Moment

$$m_{ij}' = C_{ij}(l_{x_i}\bar{l}_{x_j} + l_{y_i}\bar{l}_{y_j} + l_{z_i}\bar{l}_{z_j}). \qquad (6)$$

3. Rotation-Force

$$m_{ij}' = -C_{ij}\begin{vmatrix} l_{x_i} & l_{y_i} & l_{z_i} \\ x_i - \bar{x}_j & y_i - \bar{y}_j & z_i - \bar{z}_j \\ \bar{l}_{x_j} & \bar{l}_{y_j} & \bar{l}_{z_j} \end{vmatrix}. \qquad (7)$$

4. Translation-Moment

$$m_{ij}' = 0. \qquad (8)$$

In the above expressions the symbols l_x, l_y, etc., are direction cosines derived from the direction numbers defined in Fig. 1. The symbol C_{ij} is defined as follows:

$C_{ij} = 1$	if the ith degree of freedom contains a component of that member of the jth subordinate force pair for which direction numbers were defined.
$C_{ij} = -1$	if the ith degree of freedom contains a component of that member of the jth subordinate force pair for which direction numbers were not defined.
$C_{ij} = 0$	if the ith degree of freedom contains no component of the jth subordinate force pair.

The symbols C_{ij} are referred to as "topologic coefficients," since they define the manner in which the structure is connected. The m_{ij}' are the elements of the matrix M' thus:

$$M' = [m_{ij}'].$$

Let

$P_{0_{ij}}'$ = the component in the ith degree of freedom of the jth subordinate external load;

$P_{\Delta_{ij}}'$ = the component in the ith degree of freedom of the jth subordinate dummy (deflection) load.

$P_{x_{ij}}'$ = the component in the ith degree of freedom of the jth subordinate unit redundant force pair.

Evidently $P_{0_{ij}}'$, $P_{\Delta_{ij}}'$, $P_{x_{ij}}'$ are analogous to m_{ij}' and can be obtained by formulas similar to (5), (6), (7), and (8). These analogous formulas need not be written down. The following matrices are defined:

$$P_0' = [P_{0_{ij}}'] \qquad P_\Delta' = [P_{\Delta_{ij}}'] \qquad P_x' = [P_{x_{ij}}'].$$

The components of the principal forces in the various degrees of freedom may be obtained from the components of the subordinate forces by the following transformations:

$$\left.\begin{aligned} M &= M'K_m \\ P_0 &= P_0'K_0 \\ P_\Delta &= P_\Delta'K_\Delta \\ P_x &= P_x'K_x \end{aligned}\right\}; \qquad (9)$$

where

$K_{m_{ij}}$ = contribution to the ith subordinate force pair of jth unit force pair,

$K_{0_{ij}}$ = value of ith subordinate external load in jth loading condition,

$K_{\Delta_{ij}}$ = contribution to ith subordinate dummy (deflection) load of jth unit deflection load,

$K_{x_{ij}}$ = contribution to ith subordinate redundant force-pair of jth unit redundant force-pair.

The Deflection Influence Matrix and Free Vibration

The deflection influence matrix is given by

$$\delta = f_\Delta{}^T D[I - f(f^T D f)^{-1} f^T D] f_\Delta. \qquad (10)$$

The natural frequency ω and the modal column Δ can be obtained by standard methods from

$$(\omega^2 \delta M - I)\Delta = 0, \qquad (11)$$

where M is a mass matrix.

Some Applications of the Program

Some of the structures to which the static stress program has been applied are shown in Figs. 2 to 6. Fig. 2 is a diagram of the root structure of a swept wing. This structure had 98 elastic elements, 127 statically determinate forces, and 28 redundants. The complete stress distribution was obtained for 28 separate loading conditions. Deflections were obtained at 28 points and the deflection influence matrix at the outboard end was computed for use in flutter analysis.

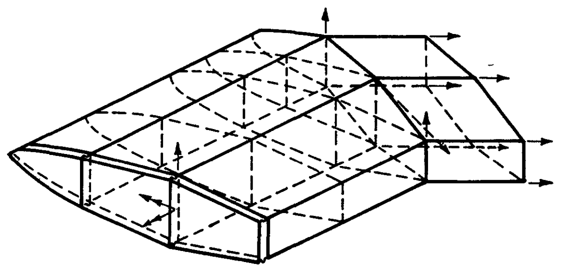


Fig. 2—Swept wing root structure.

Fig. 3 shows a doubly symmetric segment of fuselage structure having 5 frames and 18 stringers. Loads were applied to the cross beam at section AA. The purpose was to find the stresses in the center frame; the rest of the structure was included to provide a realistic condition of support. There were 180 statically determinate forces and 29 redundants. Stresses were also obtained in the adjacent frames and fuselage skin.

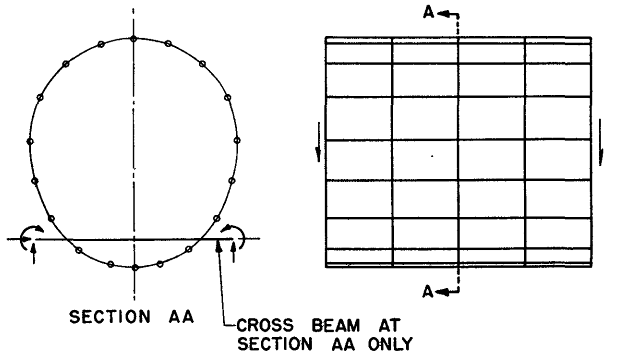


Fig. 3—Fuselage frame analysis.

Fig. 4 is a diagram of a noncircular, nonuniform fuselage frame for which natural modes and frequencies of vibration in the plane of the ring were computed. There were 24 elastic elements, 12 mass elements and 2 redundants. The solution required the determination of the eigenvalues of a 24th-order matrix. The figure shows the fifth mode of vibration. As a check on the method, the modes and frequencies of a uniform circular ring were also computed. The results checked values computed from Hoppe's formula with a maximum of 2 per cent error for the first five modes, even though only seven mass elements were used per 90 degrees of arc.

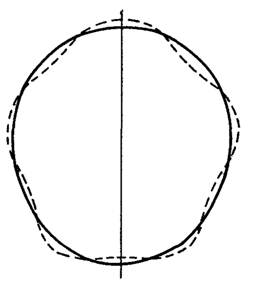


Fig. 4—Vibration of a noncircular, nonuniform ring.

A rib structure of the Vierendeel type is shown in Fig. 5. This rib was about 8 feet long, and was worthy of careful analysis. The upper part of the figure shows the structure and loading, the lower part shows the equivalent elastic framework and method of support. As indicated, support was provided by a number of springs representing the elastic restraint provided by surrounding structure. There were 118 statically determinate forces and 47 redundants. A complete stress and deflection analysis was obtained. The deflections are shown in the upper part of Fig. 5. Notice the complicated shape of the outline of the elastic framework. This complication is typical of the geometric detail which can be accounted for by digital methods.

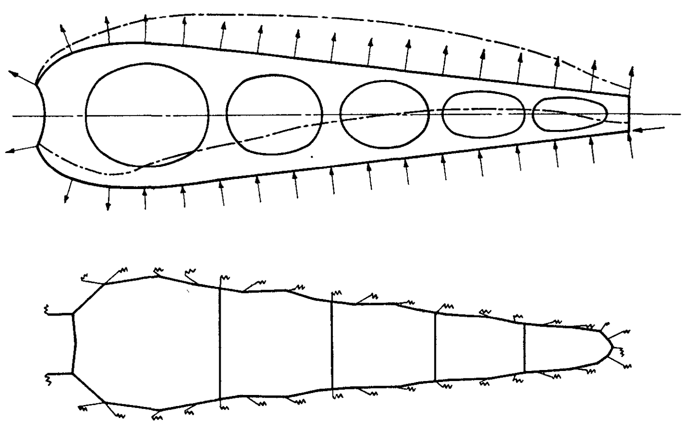


Fig. 5—Rib analysis.

Matrix Arithmetic for Systems of Large Order

The foregoing section has described a method for treating a large and important class of problems which arise in the design of airframes. It is a method with inherent appeal for its conciseness and generality, and for the explicit manner in which it exhibits the relationships existing among the physical quantities which characterize the structure.

The power of the method as a practical tool must, nevertheless, be finally measured in terms of the success with which the necessary matrix operations can be carried out when the system is one of large order. The remainder of this paper will be devoted to a description of the steps taken by the Computing Engineering group at Douglas, Santa Monica, to develop a digital computer program which is capable of performing these calculations rapidly and accurately.

The expression "a matrix of large order," is subject to varied interpretations, depending upon the circumstances in which it is encountered. It is natural that the computing analyst should assign to it a meaning which is primarily a function of the digital computer facilities at his disposal. The computer in use at Douglas, Santa Monica, is the IBM Model 701, providing 2,048 words of fast-access electrostatic memory, with auxiliary storage consisting of four magnetic drums and four magnetic tape units. Matrix operations are most conveniently and rapidly performed if they can be entirely contained within the electrostatic memory of this computer and the limiting dimensions for such operations may be put at 30×30. In what follows, the term "a matrix of large order," will imply one of dimensionality exceeding this limit; for such matrices, it is necessary at all times to make use of one or another of the auxiliary memory units.

Most of the matrix problems encountered by a group which serves as the computing facility for a large engineering department are of moderate dimensions. In order to provide solutions in the least possible elapsed time, and at small programming cost, it is necessary to make extensive use of standard matrix routines; this is especially true of jobs of an exploratory nature, and those which consist of a relatively small number of cases.

This method is often uneconomical, however, for a general program, and in particular for one requiring significant computing time for its solution. In such a case, a program more closely tailored to the characteristics of the problem may be desirable.

Programming the Static Stress Analysis

The static stress analysis program has provided the motivation for developing a specialized group of routines to perform large-order matrix arithmetic, in which the operands for a matrix operation are stored on two magnetic tapes, and the resultant is written on a third tape as it is generated. Each row or column of a matrix forms a separate record on the tape, and because only one such record need be retained in electrostatic memory at any time, it is manifestly possible to handle systems of considerable dimensions; a nominal limit of 300th order is imposed for these routines. Since many of the matrices, especially those of the equilibrium and linking equations, are characterized by a large proportion of zero elements, the principle has been adopted that only nonzero elements are retained, each with its row and column indexes.

The operation of inversion, or, more precisely, the solution of a system of simultaneous equations, occurs twice in the static stress analysis: first, in solving the equations of static equilibrium [see (3)] and second, in obtaining the redundant forces [see (1)], where the former is, in general, of much larger order than the redundant system. It is possible, however, to so order the degrees of freedom and forces that the coefficient matrix M of the equilibrium system is of triangular form, or so nearly triangular that premultiplication by an operator, which may be obtained by inspection, reduces it to this form. In such a case, the solution of the simultaneous equations is easily carried out in stepwise fashion with the matrices stored on tape.

It may be pointed out that a procedure which gives this triangular matrix is familiar to the stress analyst; in fact, it is equivalent to the method of joints, in the analysis of statically determinate pin-jointed trusses.

A solution for the redundant forces offers no such direct escape. The present program imposes a limitation of thirty or less on the number of such forces, and the inversion is accomplished in the electrostatic memory, using the well-known Jordan elimination method. However, a structure with nearly twice this number of redundants has been treated by the method of partitioning, and a modification of the program now in preparation is expected to permit consideration of as many as 110 redundants in the near future.

All matrix operations described above are carried out using standard floating binary point arithmetic; this device, although relatively slow on the Model 701, has been found indispensable in preserving the accuracy of the calculations.

The generation of the equilibrium matrices from their topologic equivalents is a recent addition to the program. It is a basically nonmatrix operation which materially reduces the amount and complexity of information which the stress engineer must provide when he prepares a problem for solution.

Computing Time

Computing times for the routines described above, when matrices are stored on tape in addressed-element form, are somewhat variable, depending upon the proportion of null elements. Approximately three hours are needed for the solution of a typical system with 225 degrees of freedom and the maximum of 30 redundant forces, including card input and printing operations. Some representative times for specific operations are given below:

Addition:	228×228 (full)	3.3 minutes
Multiplication:	$(228 \times 228)\cdot(228 \times 228)$	
	(2% full)	20 minutes
	$(100 \times 25)\cdot(25 \times 25)$ (full)	15 minutes
Inversion:	25×25 (full)	1 minute.

It is characteristic of digital techniques for large-order matrix systems that what may be called "bookkeeping" considerations assume an increasing importance. An operation such as transposition, which is trivial for a matrix retained entirely in electrostatic memory, involves significant amounts of sorting and collating when the matrix is stored on tape, and the time required quickly approaches that necessary for the multiplication of a comparable system.

Numerical Accuracy of the Method

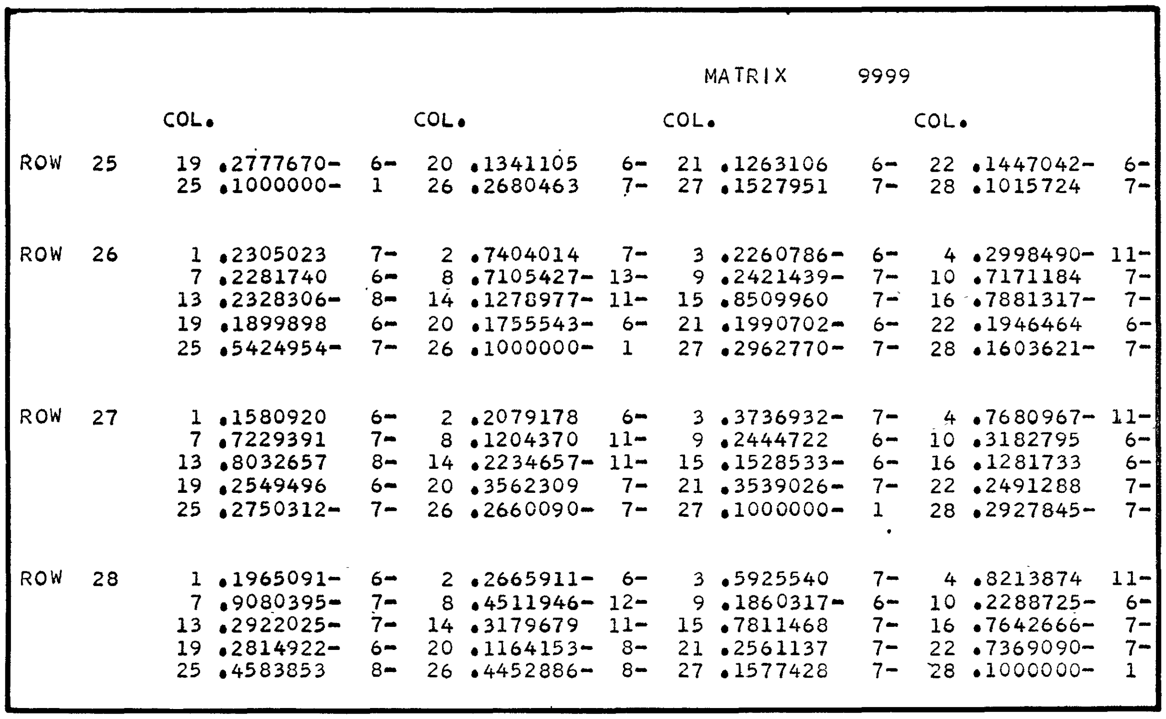
There is a considerable body of recent literature dealing with the errors which occur in matrix and other calculations involving many repeated arithmetic operations upon approximate numbers such as those used in digital computation. No attempt will be made here to review the results of these studies. The formulas which have been developed to estimate bounds for such errors are a function of the individual case, and are often difficult to apply in practice.

It is, however, possible for the analyst to ensure a satisfactory degree of computational accuracy by appealing to certain physical principles of structural analysis. First, as has been pointed out, the force pairs and degrees of freedom for the statically determinate structure should be numbered in such an order that the M matrix is approximately or actually triangular, since this procedure not only simplifies the solution but also tends to provide a strongly nonsingular matrix. The second principle may be stated as follows: If the structure which remains after relaxation of the redundant constraints is physically strong, then the solution of the continuity problem tends to be mathematically strong, because the redundants are small compared to the statically determinate forces. This fact was pointed out by Hoff[6] in connection with the analysis of a plane ring.

A check of over-all accuracy can quickly be made by verifying the equilibrium of forces acting upon individual elements chosen as a sample from those which comprise the structure.

Fig. 6 reproduces a portion of an accuracy test of the inversion performed in determining the internal forces and deflections of a structure. This test is calculated as a routine part of the analysis, and the result, which should closely approximate the negative identity matrix, is printed. The matrix elements shown consist of a characteristic part of seven significant digits, followed by the power of ten by which this characteristic must be multiplied to obtain the true number. In the matrix of order 28×28, from which the figure was taken, the largest off-diagonal element was −0.00000208, and the inversion was considered satisfactory.

An element selected at random from a wing root study



```
                                                    MATRIX      9999

              COL.                COL.                COL.                COL.

ROW   25      19 .2777670-   6-   20 .1341105    6-   21 .1263106    6-   22 .1447042-   6-
              25 .1000000-   1    26 .2680463    7-   27 .1527951    7-   28 .1015724    7-


ROW   26       1 .2305023    7-    2 .7404014    7-    3 .2260786-   6-    4 .2998490-  11-
               7 .2281740    6-    8 .7105427-  13-    9 .2421439-   7-   10 .7171184    7-
              13 .2328306-   8-   14 .1278977-  11-   15 .8509960    7-   16 .7881317-   7-
              19 .1899898    6-   20 .1755543-   6-   21 .1990702-   6-   22 .1946464    6-
              25 .5424954-   7-   26 .1000000-   1    27 .2962770-   7-   28 .1603621-   7-


ROW   27       1 .1580920    6-    2 .2079178    6-    3 .3736932-   7-    4 .7680967-  11-
               7 .7229391    7-    8 .1204370   11-    9 .2444722    6-   10 .3182795    6-
              13 .8032657    8-   14 .2234657-  11-   15 .1528533-   6-   16 .1281733    6-
              19 .2549496    6-   20 .3562309    7-   21 .3539026-   7-   22 .2491288    7-
              25 .2750312-   7-   26 .2660090-   7-   27 .1000000-   1    28 .2927845-   7-


ROW   28       1 .1965091-   6-    2 .2665911-   6-    3 .5925540    7-    4 .8213874   11-
               7 .9080395-   7-    8 .4511946-  12-    9 .1860317-   6-   10 .2288725-   6-
              13 .2922025-   7-   14 .3179679   11-   15 .7811468    7-   16 .7642666-   7-
              19 .2814922-   6-   20 .1164153-   8-   21 .2561137    7-   22 .7369090-   7-
              25 .4583853    8-   26 .4452886-   8-   27 .1577428    7-   28 .1000000-   1
```

Fig. 6—Portion of inversion check. $-I = -(f^TDf)^{-1}(f^TDf)$.

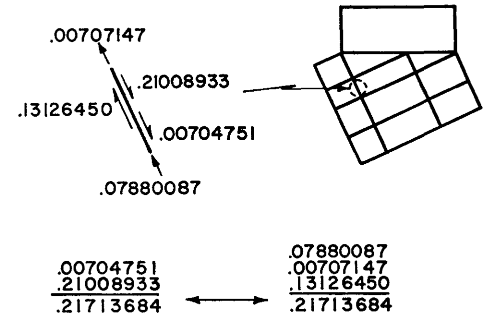


Fig. 7—Equilibrium check.

[6] N. J. Hoff, "Stress analysis of rings for monocoque fuselages," *Jour. Aeronaut. Sci.*, vol. 9, p. 245; May, 1942.

is presented in Fig. 7, showing the forces which act upon it, as well as a numerical verification that the equilibrium conditions are satisfied.

It is perhaps superfluous to add that the final test of results obtained with any such mathematical model must be their reasonableness, based upon experience with the behavior of similar structures and upon experimental verification whenever this is possible.

Conclusions

A general digital-computer program for static stress analysis has been described. This program is applicable to a large class of structures without additional programming time, and relieves the stress analyst of most of the computational burden. After the analyst has idealized the structure, and has made it statically determinate, his task consists of a straightforward tabulation of elastic and geometric properties. Once a unit load solution has been obtained, multiple load conditions can be investigated at little additional cost, and the effect of varied elastic properties can be studied by rerepeating only the solution of the continuity equations.

Extensive use of magnetic tapes makes possible the large-order matrix operations characteristic of the stress analysis problem. The program is designed throughout for economy of computation.

Aircraft Performance Studied on an Electronic Analog Computer

L. B. WADEL AND C. C. WAN[†]

Introduction

PERFORMANCE analysis of an airplane consists of the study of various operating characteristics and operating limits. These characteristics may be divided essentially into three major groups as follows:

1. *Performance at Constant Altitudes.* These include the conventional performance characteristics such as maximum speed, rate of climb, and equilibrium turn performance. All of these items may be determined by comparison of "thrust available" and "thrust required" for steady-state operations, based on constant airplane weight and constant altitude.

2. *Performance at Varying Altitudes.* For certain portions of a typical flight program, fuel economy and elapsed time become factors of major importance. It is necessary to establish a theoretical optimum operating schedule which would serve as an upper bound to the attainable performance. Optimum-climb programs for an interceptor and optimum-cruise profiles for a long range bomber belong to this category. Although these problems may be rigorously resolved by means of the calculus of variations, equivalent methods requiring only steady-state considerations can be formulated to yield the required solutions.

3. *Range Performance.* The maximum range, or radius of action, of an airplane for a specific mission profile is one of the most important items in performance analysis. Here, the variation of operating altitudes and airplane weight must be properly accounted for. Generally, this can be achieved through appropriate combination of results obtained from the first two groups.

[†] Chance Vought Aircraft, Inc., Dallas, Tex.

During the preliminary design of a prototype aircraft, a fairly exhaustive study of range performance is usually carried out for a range of design parameters. The proper selection of compromises among these design parameters is made on the basis of the results obtained from such a study. Similar performance calculations would be made also for guided missiles, but different properties will be emphasized. For production aircraft, performance data must be presented in the pilot's handbook. A wide range of variations in atmospheric conditions and mission types is usually included in this presentation.

It may readily be realized that a large amount of computational effort is necessary to provide a complete assessment of the pertinent performance characteristics of an aircraft. It is, therefore, desirable to mechanize performance calculations as much as possible. Several exploratory analog computer studies have been made, with promising results, of certain performance characteristics of a high speed fighter aircraft. The method of approach and the nature of the results obtained during these studies are described in this paper.

The Electronic Analog Computer

The general-purpose electronic analog computer (electronic differential analyzer) as we know it today was developed in the years immediately following World War II; its original mission was to solve the ordinary differential equations which describe the dynamic behavior of guided missiles. The key to such computers is the operational or computing amplifier. This is a high-gain dc amplifier which, when supplied with proper precision input and feedback impedances (usually resistors and capacitors), performs summation, integration, sign-changing, or other more specialized linear operations.

Other computer components are potentiometers, used to establish constant coefficients and additive constants, and such nonlinear devices as multipliers and arbitrary function generators. For a given problem, these basic "building blocks" are interconnected such that the mathematical equations of the electrical behavior of the circuits so established are identical with the equations corresponding to the aerodynamic, mechanical, chemical, pneumatic, hydraulic, economic, biological, electrical, or electronic system or process that is under investigation. At any time the output of a given unit in volts represents the value of a variable, or a function of a variable or combination of variables in the system being studied. Direct-recording oscillographs provide time-histories of selected outputs.

For most problems solved on electronic analog computers, nonlinearities, although often important, are few in number. Attention is focused upon relatively small dynamic oscillations about a predetermined set of equilibrium conditions. The order of the system of differential equations being solved determines the number of integrators. For example, the linearized equations for the control-fixed longitudinal motion of an aircraft require four integrators. The number of summers and sign-changers, as well as potentiometers, depends upon the number and signs of the terms in the equations. Perhaps 30 per cent of the amplifiers employed will be integrators.

As noted earlier in the paper, we are concerned here with the solution of problems of special concern to the aircraft designer. Obviously, we must know what equilibrium flight conditions are possible or desirable for an aircraft or missile before it makes sense to discuss dynamic performance about a certain flight condition. As far as the computer is concerned, what is special about performance equations? Qualitatively, nothing. The same type of computer components are required, but the emphasis is drastically shifted. The percentage of integrators required may drop to 5 or less, but multipliers and function generators are required in profusion. The many nonlinearities in the equations assume fundamental importance, and their accurate representation in the computer becomes the prime difficulty in solving the equations.

Input Data

Basic data required for performance analysis may be grouped into four general categories as follows:

1. *Aerodynamic Data.* These include the lift and drag characteristics. The drag coefficients, being directly related to thrust required, require careful simulation. These coefficients are functions of both Mach number and the lift coefficient.

2. *Power Plant Data.* These include the installed thrust and corrected fuel rate for a specific power plant installation. These coefficients are concerned with available thrust and fuel rate which are functions of altitude, Mach number and power setting. A very elaborate simulation setup is usually required for these coefficients.

3. *Atmospheric Data.* These include the air density and the speed of sound; they are functions of altitude only.

4. *Geometrical and Weight Data.* These are essentially constant factors for a particular design. However, for problems involving the consideration of the entire flight profile, the fuel rate and other step variations due to jettisoning of bombs or external stores must be taken into account.

The general approach to the simulation problem is to establish a polynomial approximation to the basic data in terms of the most significant independent variables. By this means, functions of more than one variable may be simulated by a combination of function generators and multiplying servomechanisms. For example, the drag coefficient C_D of an airplane may be expressed in terms of the lift coefficient C_L as

$$C_D = d_0 + d_2 C_L^2 + d_4 C_L^4,$$

where d_0, d_2 and d_4 are all functions of Mach number and may be stored on such function devices as photoformers, diode function generators, or nonlinear potentiometers.

Constant Altitude Performance

Performance characteristics of an airplane at constant altitude is related to the difference between the available thrust (T) and the total airplane drag (D). It is, therefore, necessary to generate by appropriate function devices these two quantities for various operating conditions. The range of simulation, of course, should be sufficiently extensive to include all conceivable altitudes and speeds.

The available thrust is a function of both altitude and speed for a specific airplane-power plant combination. The airplane drag may be expressed in terms of the airplane drag coefficient (C_D) as

$$D = q^* S M^2 C_D, \qquad (1)$$

where q^* is the dynamic pressure at sonic speed and a function of altitude, S is the wing area, M is the Mach number and C_D is further related to the airplane lift coefficient (C_L) by

$$C_D = d_0 + d_2 C_L^2 + d_4 C_L^4. \qquad (2)$$

The airplane lift coefficient is related to the airplane weight (W) and the airplane load factor (n) by the equation

$$nW = q^* S M^2 C_L. \qquad (3)$$

For constant altitude performance calculations, the airplane weight is usually treated as a constant.

1. *Equilibrium Turn Performance.* During equilibrium turns, there is to be no loss in speed and altitude. The controlling equation is therefore

$$T - D = 0. \qquad (4)$$

The airplane load factor (n) corresponding to this condition is of considerable interest for fighter type aircraft, as it provides an index to the maneuverability of the aircraft. Furthermore, the maximum and the minimum speeds for level flight may be defined by means of (4) by setting the airplane load factor equal to one.

A feedback circuit may be used for equilibrium turn computation by treating the quantity $(T-D)$ as an error signal input to the C_L channel. An integrator may be used to provide a variable Mach number input so that the final turning performance curve may be recorded directly on a plotter. Typical results of this calculation are displayed in Fig. 1.

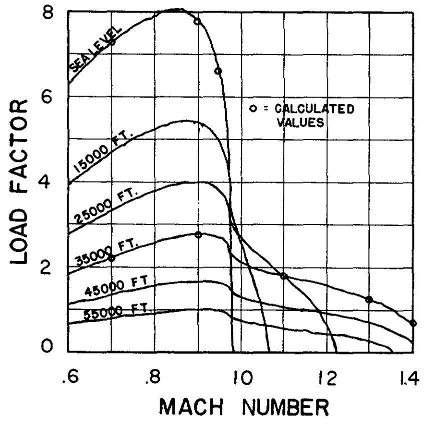


Fig. 1—Equilibrium turn performance of a typical fighter airplane.

2. *Acceleration Characteristics.* The time required to reach a given speed and the variation of speed during a turning maneuver are required in the study of combat situations. Depending on the magnitude of the airplane load factor during the turn, the quantity $(T-D)$ may be either positive or negative. The equilibrium turn discussed above defines a limiting case where this quantity $(T-D)$ is zero. When the airplane load factor is higher than the equilibrium turn load factor, at the same Mach number, the airplane would decelerate. Acceleration is only possible when the airplane load factor is lower than the equilibrium turn load factor.

For operations at constant altitudes, it is convenient to study only turn at constant load factors. The basic equation for this case is

$$(Wa/g)\dot{M} = T - D, \tag{5}$$

where a is the speed of sound at the given altitude and g is the gravitational acceleration. The constant load factor under consideration may be used as a limiter for the generation of the C_L signal. The time history may be obtained through integration of (5). Typical results of this problem are displayed in Fig. 2.

3. *Rate of Climb.* The classical treatment of rate of climb is based on constant speed operations at fixed power setting. Under these conditions the rate of climb ($\dot{h}$) may be expressed as

$$\dot{h} = (T - D)Ma/W, \tag{6}$$

when a is the speed of sound and a function of altitude. The airplane drag term in (6) must be evaluated for a lift coefficient (C_L) defined by

$$C_L = C_{L_\alpha}[W - T(\alpha_0 + e)]/[T + q^*SM^2C_{L_\alpha}]. \tag{7}$$

In (7), C_{L_α} is the slope of the trimmed lift coefficient curve with respect to the angle of attack, α_0 is the angle of the zero lift line and e is the thrust angle. Both C_{L_α} and α_0 are usually functions of Mach number while the thrust angle is a constant for a specific airplane.

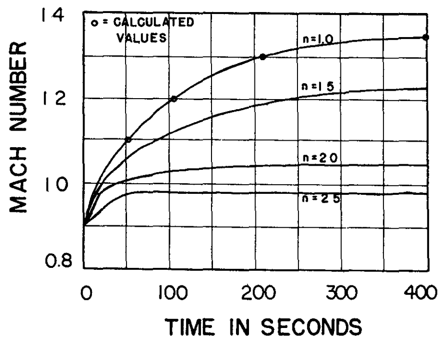


Fig. 2—Acceleration characteristics of a typical fighter airplane.

The rate of climb may be obtained by direct computation by means of only function devices and appropriate circuitry. An integrator may be included to provide a variable Mach number input and the final rate of climb at constant altitude may be recorded directly on a plotter. The best-climb speed and the best-climb Mach number may be obtained from this plot. Typical examples of rate of climb calculations are displayed in Fig. 3.

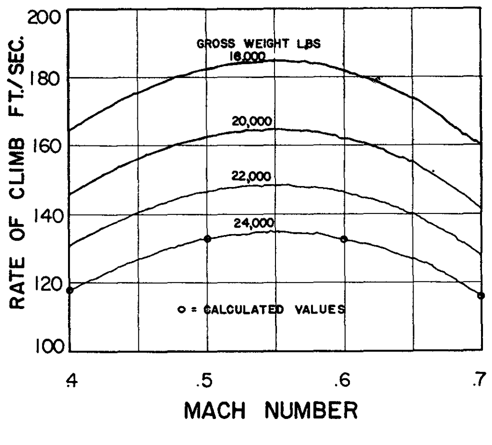


Fig. 3—Rate of climb characteristics of a typical fighter airplane.

Performance at Varying Altitudes

In addition to the constant altitude performance discussed above, it is frequently desirable to establish optimum operating schedules for various portions of a flight plan. Typical among these are the optimum climb profile and the optimum cruise profile. The optimization in these cases may be made on the basis of either fuel economy or elapsed time. Although these problems could be formulated rigorously by means of the calculus of variations, solutions are not always attainable due to mathematical complexities. Equivalent methods requiring only steady-state considerations are available, whereby the solutions may be obtained by graphical means. Certain parametric curves must be constructed, however, before proceeding with the graphical solution.

The types of parametric curves required are listed in Table I for both optimum climb and optimum cruise calculations. A discussion of the significance of these curves may be found,[1] but is not included here for the sake of brevity. Additional function devices are required to generate the fuel rate input (w_f) for these calculations.

TABLE I

Parametric Curves for Performance Calculations

Optimum Climb	Minimum Time	$\dot{h}=(T-D)Ma/W$ $=$constant	$h-M$ plane
	Minimum Fuel	$\dot{h}/w_f=$constant	$h-M$ plane
Optimum Cruise	Maximum Time	$h=$constant	w_f-M plane
	Minimum Fuel	$h=$constant	Ma/w_f-M plane

For optimum climb calculations, the pertinent factors may be maintained at a constant value by using the deviation from the constant as a feedback to the altitude channel. For optimum cruise calculations, direct computation is possible and no special technique is required.

Range Performance

The calculation of maximum range, or radius of action, of an aircraft requires the consideration of the complete profile necessary for the accomplishment of a specific mission. Fig. 4 shows schematically the various

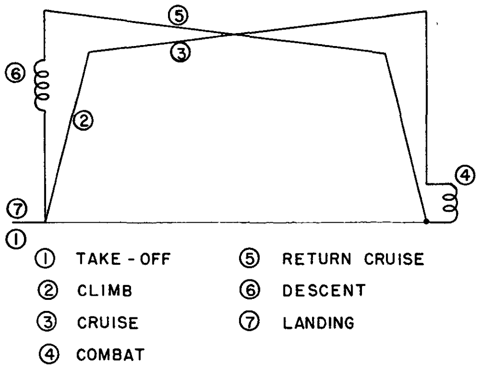


Fig. 4—Typical mission profile.

phases of flight: take-off, climb, cruise, combat, return cruise, descent, and landing. The characteristics of certain phases will be determined by the specific mission, while others may be typical of the particular aircraft, regardless of mission. An initial guide to the other phases can be obtained from the results of previous performance calculations. Because of the complexity of the equations and the number of phases of flight, radius of action may be computed most easily by an iterative procedure. This method may be illustrated by the following much-simplified example:

Consider a flight to consist only of a straight-line cruise-to-target phase plus an immediate return-cruise phase, all at a constant speed and altitude (no wind), with an instantaneous discharge of a weight w over the target. The following equations then apply: (for a typical airplane)

$$W = 10^5 C_L \text{ lb.} \tag{8}$$

$$T = 10^5 C_D = 10^5(0.015 + 0.15 C_L^2) \text{ lb.} \tag{9}$$

$$\dot{W} = -w_f = +486 - T \text{ lb/hr.} \tag{10}$$

$$x = Vt. \tag{11}$$

Given:
$W(o) = 30{,}000$ lb.
$w =$ bomb weight $= 2{,}000$ lb.
$W(e) =$ minimum landing weight $= 20{,}000$ lb.
$V =$ speed $= 500$ knots

A possible iteration scheme is as follows: Choose a trial radius of action R_0, corresponding to a time-to-target, t_0. Carry out the computations outlined above, stopping when $t=t_0$. Reduce W by w, then continue until the allotted fuel is exhausted. The total distance covered will be $X_0=R_0+r_0$, where r_0 is the return distance traveled from the target. We seek a radius of action R such that $r=R$, or $X=2R$; i.e., fuel exhausted exactly upon return to base. To find this radius of action, we choose $R_1=X_0/c$ (in general, $R_{k+1}=X_k/c$), and repeat the solutions until sufficient convergence is achieved. It is possible to make such iteration automatically handled by an electronic analog computer.[2] Fig. 5 shows the suc-

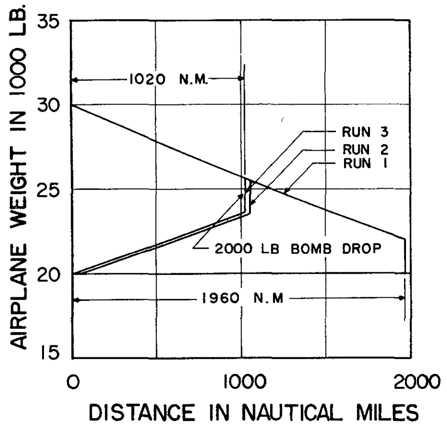


Fig. 5—Iterative solution of a range problem.

[1] E. S. Rutowski, "Energy approach to the general aircraft performance problem," *Jour. Aeronaut. Sci.*, vol. 21, pp. 187–195; March, 1954.

[2] L. B. Wadel, "Automatic Iteration on an Electronic Analog Computer," 1954 Western Electronics Show and Convention; August 25, 1954.

cessive results obtained for the sample problem just discussed. In this case an analytical solution exists, yielding $R=1{,}025$ nautical miles.

To handle actual radius of action problems, it would be desirable to incorporate in the computer automatic switching between successive phases of a flight plan, as well as a more sophisticated automatic iteration technique.

Conclusion and Future Plans

Several exploratory electronic analog computer studies have been made of certain performance capabilities of a modern, high-speed fighter aircraft; the method of approach and the nature of the results obtained for a theoretical aircraft have been described. A longer-range program is under study which would make available sufficient computer equipment, particularly function generators, and develop applicable techniques such that complete missions may be studied at once. A compromise is thus sought between special-purpose computing equipment that cannot easily be used for other purposes, and the general-purpose electronic analog computer whose normal complement of computer elements is not chosen for maximum efficiency in solving performance problems. It is believed that considerable advantages of speed and convenience may thus be achieved in the preliminary design of aircraft or missiles, and in the later tabulation of data for the pilot's handbook.

Acknowledgment

It is a pleasure to acknowledge the general assistance of C. C. Calvin, B. B. Mackey, Mrs. Olive Warram and Mrs. Evan Cotten, as well as the assistance of Mmes. Wilma Drake and Martha Cohen in typing the manuscript and Mrs. Elizabeth Bird in executing the figures.

Coding a General-Purpose Digital Computer to Operate as a Differential Analyzer

R. G. SELFRIDGE[†]

Summary—**This paper describes a system of coding a general-purpose digital computer so that differential equations may be solved easily and rapidly. While originally developed for an IBM 701, the system is applicable to any of several large computers, in that at no stage is any reference to a particular machine needed.**

The process is such that any engineer who has learned how to set up an analog computer should be able to code without hesitation, and should have very little difficulty even if no prior computer experience exists.

No attempt has been made to solve partial differential equations. The present system could be used, but would be inefficient in all but the simplest problems.

Introduction

THE SYSTEM TO BE described in this paper was set up specifically to enable problems which were to be run on a REAC analog computer to be run instead on an IBM 701 computer. The reason for this was twofold. First, that larger problems could be handled by a digital machine than was possible on the available analog computer, and secondly, that greater accuracy was possible. To some extent, therefore, there is a bias towards these two machines. Nevertheless, the basic ideas can easily be used on any computer with consequent advantages.

Nearly all the tricks available to an analog user can be duplicated with extreme ease, and in many cases the desired result is available without resorting to such tricks.

† USNOTS, Inyokern, Calif.

Since presently available digital computers operate serially, that is, only one operation is performed at a time, the solution of the equations will proceed step by step, one function at a time. Each equation must therefore be broken down into all its parts, and each part developed as needed. In the course of this breakdown each function is assigned a location, an amplifier number for an analog computer, or storage address for the digital computer. For the present coding there is no need or advantage in knowing where the actual storage takes place in the computer.

After assigning the storage locations (amplifier numbers), the coding for each location must be set up. For the digital computer this consists of a series of numbers; for the analog computer it is a wiring diagram and actual wiring of a patch board.

What follows is a brief description of the coding that the digital computer will accept and then a comparison of the advantages and disadvantages with an analog computer. The system as presented here has been changed slightly to eliminate a few details that are tied to the specific machine and dependent primarily on the method by which the system is constructed. One such case is the permissible range of multiplication constants.

Description

The operations that are available consist of addition, multiplication, division, integration, and a relay operation. Each of these operations is assigned a number, and also requires the listing of the storage locations of each input and an associated multiplicative constant. The relay operation is for the purpose of changing the structure of a problem during running. Such changes will cover, for example, changing the rate of integration, or blind-spot functions in control systems.

With each of these operations there are a few limitations to prevent the magnitude of the functions from getting too large. With the present coding all variables are considered as ten-digit numbers less than one, and all operations round off to ten digits.

Integration

This is given the number 1 or 2. If called for by 2 then an upper and lower limit are placed on the output. There may be as many inputs as desired, each input being given by a storage location and a multiplicative constant. This constant can be plus or minus, and multiplies the input. The last two numbers that will be listed, will be, in case operation 2 is called, the upper and lower limits desired.

In this coding system Simpson's Rule is used for integration, but any other method could be fitted if desired. The mesh size of the integration must be stored in some location, and this location is also fed in as one of the necessary inputs. This latter information would not be necessary except that it was thought desirable to be able to change the mesh size at will, and hence its location must be spelled out.

Addition

Operation 3 or 4. This operation is similar to integration in its inputs and forms the sum of the inputs. If operation 4 is called then the output will be limited between two specified numbers.

Multiplication

Operation 5. Two storage locations and a multiplicative constant must be supplied.

Division

Operation 6. The location of the dividend and a multiplicative constant must be supplied, and the location of the divisor.

Relay

Operation 7. This operation affects any part of the program that is desired, and changes operations, input locations, or multiplicative constants as a function of the sign of the difference of two inputs. While this particular operation has proved valuable, its form is dependent on the specific coding and machine used. Rather than describe its form as used for the 701, its purpose will be given. With this relay operation any effect that is possible with relays on an analog computer should be possible. This includes such effects as blind spots in functions, or step functions. It is also possible to change the rate of integration, by changing the mesh size used. It is easy to arrange an absolute value function, or automatic scaling on parts of a problem, though the latter effect is very extravagant in machine time and space.

Two extra operations need to be mentioned. One, operation 8, stops the problem when some predetermined condition is satisfied, and the other, operation 9, is only for indicating the completion of problem coding.

Since computation proceeds serially, the storage locations are assumed to be consecutive, starting at 1, and the coding to be assigned to each location is also assumed to be consecutive.

As a brief indication of the system planned, let us consider a very simple differential equation, and its coding, both for the digital and the analog computer.

$$\frac{d^2y}{dx^2} + 1/2\frac{dy}{dx} + y = x, \qquad \dot{y}(0) = 1/10, \qquad y(0) = 0.$$

In order to maintain the comparison as far as possible, let us assign dy/dx storage location 1, and amplifier 1 (with the feedback appropriate to an integrator), y storage location 2 and amplifier 2, and x storage location 3, and amplifier 3. The extra constants necessary for the digital machine are the mesh size in storage location 4, and in storage location 5 a constant 1, necessary in generating x. With these definitions the analog computer diagram might be as shown in Fig. 1. The digital computer coding for the same problem would be as shown in Table I, where L is used to indicate a storage location.

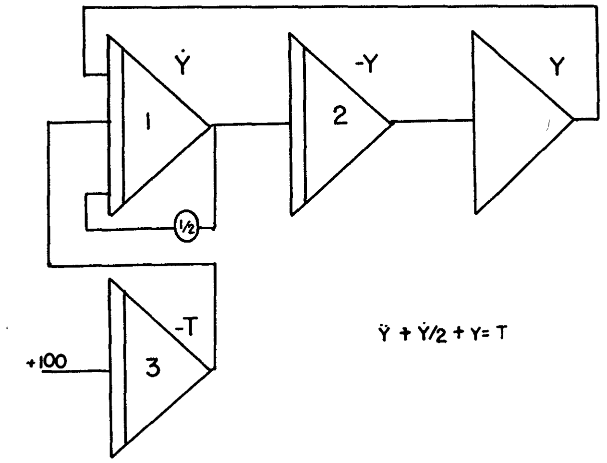


Fig. 1

If this coding is followed through, it is clear that it is very similar to the analog coding. The initial conditions are inserted by punched cards (or other standard means), as is the problem coding. The type and quantity of output is too dependent on the particular machine to be described. It must suffice to mention that graphical output is available with some computers, and that it is simple to arrange for almost any type of output.

The advantages and disadvantages of this system can be covered easily. The main disadvantage at present is speed of operation. Given a problem completely set up on an analog computer, the running time will be considerably less than for the digital machine. This is, however, curable to the extent that digital machines will continue to speed up. There may also be objections on a specific machine because of lack of desired output form, but this can be remedied.

On both computers there will be the trouble of scaling, to prevent variables running beyond assigned limits. This seems incurable on an analog machine, whereas new digital computers with built in "floating point" operation will remove this problem. Such computers are beginning to become available; the only reason the present system does not operate as "floating point" is the time loss involved in such arithmetic.

Changing initial conditions or problem constants ("pot" settings) is equally easy in either machine, since the digital machine can take any input desired.

There are, however, several advantages for the digital computer that are of considerable importance.

First of all, the digital computer has a far greater accuracy available. Since ten digits are carried, one can expect five-digit accuracy or better, even on very large problems. This can also be an aid in scaling since division of all the variables by a large constant will prevent overloading, and still not lose accuracy.

Secondly, one can run problems that are not practical on an analog computer because of size limitations or scale factor limitations (where the range of a variable is too great). Problems requiring more multiplications than the analog can handle can be programmed easily, since there is no difference as far as the digital computer is concerned between the ease of integration, addition, or multiplication. For the digital computer the limit on the size of the problem is determined primarily by the operating time that is permissible, hence very large problems can be handled if low accuracy is possible, or enough machine time is available.

One other advantage that has, in practice, become one of the most important, is the ease of setup. Comparable problems can be coded and run on a digital machine far faster than on an analog computer. This is partially because of the ease of coding, with a wider range of input multipliers, and partially because wiring a complicated patch board can be a major undertaking of itself, and there is no comparable step in the digital system. A brief example of this was a problem in aeroballistics where one hour's work, including half-an-hour of computer time, was sufficient to add another equation on the digital machine, while about six hours were necessary on the analog machine.

Conclusion

If one considers present-day machines, the digital computer will solve many problems that an analog machine cannot; it can yield better accuracy, and elapsed time from start to finish can be materially shortened on all but very simple problems. Certain operations are also possible that are not possible on an analog computer.

Initial conditions and constant settings can be changed with equal ease in both machines.

The analog machine, when set up, will operate faster than the digital machine, and, at present, is cheaper to operate than the large digital machine needed (this is only on a per hour basis; over-all time may more than compensate).

While there are very definitely certain types of problem that are more suited to an analog machine, many of the problems run at present could be transferred, with advantage, to digital computers.

TABLE I

	1	L	4	Operation 1, with Δx in L4	yields dy/dx
L	1		$-\frac{1}{2}$	Input $-\frac{1}{2}\,dy/dx$	
L	2		-1	Input $-y$	
L	3		1	Input x	
	1	L	4	Operation 1, with Δx in L4	yields y
L	1		1	Input dy/dx	
	1	L	4	Operation 1, with Δx in L4	yields x
L	5		1	Input 1	
	9			Operation 9, coding complete	

Location	1	$d\dot{y}/dx$
	2	y
	3	x
	4	Δx
	5	$\frac{1}{2}$

Introduction to Session on Learning Machines

WILLIS H. WARE†, SESSION CHAIRMAN

THE FOLLOWING four papers treat various levels of complexity in machine systems which exhibit learning ability. It is perhaps a misnomer to label such devices as "learning machines," since it is not the machine itself, but the program that guides the machine toward its goal, which learns. This topic is unfortunately suggestive of the considerable publicity that has attended the "Giant Brain" ballyhoo directed toward digital systems, but it is not the intention of these four papers to indicate how to make "thinking" machines. The first effort in this direction is a careful definition of "thinking," which no one has yet given. It should be very carefully emphasized that machines may be said to "think" only to the extent that their users and designers have been able to penetrate sufficiently deeply into a problem so that all possible situations which might arise have been examined, and an adequate program written which describes to the machine an unambiguous criterion for its use in each case. Machine operation which results from such subtle programs is then an imitation or simulation of operations which have previously been thought through and established by human beings.

It is, however, the intention of these papers to describe certain experiments in which machine systems imitate some of the self-organizing and learning processes of the nervous system. Whether such simulations are true in the fine structure is of course unknown; much of the work described by these papers is sufficiently experimental that it is not even known that the simulation is completely correct functionally. The investigation of a process so subtle and complex as learning is not easy; the approach is by way of models which are set within the framework of a digital computing system. These papers do not suggest that future learning machines should be built in the pattern of the general-purpose digital computing device; it is rather that the digital computing system offers a convenient and highly flexible tool to probe the behavior of the models.

This group of papers deals with three distinct levels of complexity in learning systems. There is considerable similarity among the three, but the responses expected in each case and the questions asked of each system vary widely in detail. The paper by Farley and Clark deals with the most primitive and least complex level of learning. The system discussed by them is required to distinguish the simplest kind of decision and is rewarded for its correct responses. Input to the system is very restricted and there is no *a priori* set of operations which the system performs on its input. The two papers of Selfridge and Dinneen treat the next level of complexity. The input is now well organized in two spatial dimensions and exhibits a well defined topology. There is an established set of transformations for operating on this input data, although the criteria for choosing any sequence of such transformations are not *a priori* known. The paper by Newell treats the highest level of complexity—the game. Here there is a multitude of input data which must be inspected at many levels. There is a complex set of rules for interplay of this data but again the criteria for decision are *a priori* unstated.

As has been suggested by Selfridge, these levels of complexity may be likened respectively to the initial organizational efforts of neural nets in the earliest days of evolution (Farley-Clark), the learning of pattern recognition by a maturing individual (Selfridge-Dinneen), and the action of an entire man as he copes with the complicated problems of his life (Newell).

This group of papers suggests directions of improvement for future machine builders whose intent is to utilize digital computing machinery for this particular model technique. Speed of operation must be increased manyfold; simultaneous operation in many parallel modes is strongly indicated; the size of random access storage must jump several orders of magnitude; new types of input-output equipment are needed. With such advancements and the techniques discussed in these papers, there is considerable promise that systems can be built in the relatively near future which will imitate considerable portions of the activity of the brain and nervous system.

The subject matter of these papers extends over into several closely related fields. Accordingly, following the formal papers are invited comments from Walter Pitts, Research Laboratory for Electronics of the Massachusetts Institute of Technology, and George A. Miller, Associate Professor of Psychology, Harvard University. These two men have trained, respectively, as mathematical neurophysiologist and psychologist.

† The RAND Corporation, Santa Monica, California.

Generalization of Pattern Recognition in a Self-Organizing System*

W. A. CLARK† AND B. G. FARLEY†

Summary—**A self-organizing system reported upon earlier is briefly described. Two further experiments to determine its properties have been carried out. The first demonstrates that self-organization still takes place even if the input patterns are subjected to considerable random variation. The second experiment indicates that, after organization with the usual fixed patterns, the system classifies other input patterns statistically according to a simple preponderance criterion. Significance of this result as a generalization in pattern recognition is discussed. Some remarks are made on methods of simulation of such systems and their relation to computer design.**

Description of Self-Organizing System

IN A PREVIOUS paper[1] the authors described a system which organized itself from an initially random condition to a state in which discrimination of two different input patterns[2] was accomplished. The behavior of the system was simulated by means of a digital computer—the Memory Test Computer of Lincoln Laboratory.

Briefly, the self-organizing system was composed of two parts. The first part received input patterns and transformed them into outputs, and the second part acted upon parameters of the first so as to modify the input-output transformation according to certain fixed criteria. These parts were termed the transformation and the modifier, respectively.

The transformation is a randomly interconnected network of nonlinear elements, each element having a definite threshold for incoming excitation, below which no action occurs, and above which the element "fires." When an element fires, its threshold immediately rises effectively to infinity (it cannot be fired), and then, after a short fixed delay, falls exponentially back toward its quiescent value. Furthermore, at some short time after firing, an element transmits excitation to all other elements to which it is connected. The effectiveness of the excitation thus transmitted to a succeeding element is determined by a property of the particular connection known as its "weight." In general, there will be several incoming connections at any element, each having its individual weight as shown in Fig. 1. At the instant of transmission (which is the time of impulse arrival at the succeeding element), the appropriate weight is added to any excitation already present at the succeeding cell. Thereafter the excitation decays exponentially to zero. If at any time this excitation exceeds the threshold of the succeeding element, the element performs its firing cycle and transmits its own excitations.

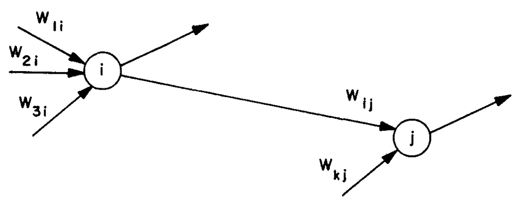


Fig. 1—Typical network elements i and j showing connection weights w.

A network such as the one described is suggestive of networks of the nerve cells, or neurons, of physiology, but since the details of neuron interaction are as yet uncertain, it cannot even be said that the networks are identical without some simplifications which are present.

In the work mentioned, the network was activated and an output obtained in the following way. The net was divided arbitrarily into two groups, designated as input and output groups. The output group was further subdivided in two, and an output was defined at any instant by the difference in the number of elements fired in the two subgroups during the instant. This arrangement might be termed a push-pull output.

The input group was also subdivided into two subgroups, and two fixed input patterns were provided, usually designated as p_1 and p_2. Input p_1 consisted in adding a large excitation into all the input elements of one subgroup simultaneously and repetitively at a constant period, but doing nothing to the other subgroup. Input p_2 was just the reverse. In this way output activity characteristic of the input pattern was obtained.

It was now desired to provide a modifier acting upon parameters of the net so as to gradually reorganize it to obtain output activity of a previously specified characteristic, namely, that patterns p_1 and p_2 would always drive the output in previously specified directions. In our experiments, p_1 was made to drive the output in a negative direction, that is to say, p_1 causes more firing to take place on the average in the first output subgroup than in the second. In the case of p_2, the situation was exactly reversed.

This desired organization of the net was accomplished by means of varying the weights mentioned above in the following way. Examination is made of the change in output at every instant. If a change in a favorable direction occurs (e.g. negative change in case p_1 is the input

* The research reported in this document was supported jointly by the Army, the Navy, and the Air Force under contract with the Massachusetts Institute of Technology.

† Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, Mass.

[1] B. G. Farley and W. A. Clark, "Simulation of self-organizing systems by digital computer," *Trans. IRE*, vol. PGIT-4, pp. 76–84; September, 1954.

[2] In this paper, the word "pattern" is synonymous with "configuration."

pattern), then all weights which just previously participated in firing an element are increased. If, on the other hand, the change was unfavorable, those weights are decreased. In our experiments the weights have values between 0 and 15 inclusive, and changes are made one unit at a time.

It is important to note that there is no detailed examination of the internal activity of the net. As a result, some of the weights may be altered in the wrong direction at any given time. However, as our results show, in the long run an over-all favorable result occurs, due to what has been termed "statistical cooperation."

Two refinements were added to the system which allow somewhat improved operation. The first is to control the level of activity in the net by manipulating a bias parameter added to all thresholds. Before each input run, the bias is set at a high value, so that no activity occurs, and it then lowers to a point somewhat below that at which activity begins. A second refinement is to add random noise to the weight sums. This tends to break up any short period activity which may occur and which may "stall" the modifier action. In other words, favorable modes of activity are introduced which may not otherwise occur.

In order to automatize the "training" of the net, the system is arranged so that if at any time the output is different from zero, inputs are automatically given as follows: p_1 if the output is positive, and p_2 if it is negative. An exception to the foregoing is made for a small "dead" zone on either side of zero in which no input is given, thus allowing activity to die out on return to zero. "Training" is then given by artificially forcing the output from zero, waiting until its return, and then forcing it to the other side until return, etc. This process is kept up until organization is satisfactory. Fig. 2 shows such a sequence having output plotted vertically with center zero, and time horizontally. It can be seen that return to zero gradually improved until it was almost unhesitating. (Unfortunately, halation due to camera and 'scope face obscures most of the detailed movement.) This net was more difficult to train than most. Occasionally a net is seen which is not successful in training. This is not surprising since the deviation in the random connection set-up is considerable.

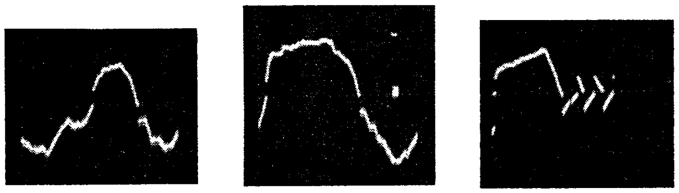

(a) (b) (c)

Fig. 2—Output record of 16-element net during organization. Input Patterns p_1 and p_2.

Purpose of Present Experiments

In the earlier work reported, only fixed input patterns p_1 and p_2 were used, as described above. However, if such a system always required precisely the same input pattern, its use would be considerably more restricted than if some variation were allowable. Therefore it is of interest to know whether the same type of organizing response can be obtained in input patterns which are caused to vary, particularly those which may contain a random variation.

Furthermore, consider the set of all possible input patterns to a net such as we have described. If the input period remains constant, there are 2^n such input patterns, each of which may be represented as a binary number of n bits, where n is the number of elements in the input group. For example, the patterns p_1 and p_2 which we have defined may be written as 11110000 and as 00001111, respectively, for a net with 8 input elements. Now, one of our nets will classify the 2^n input patterns into three classes which may be designated (+), (−), and (0), according to whether the input drives the output roughly upward, downward, or neither in a fixed time interval. An unorganized net may be expected to effect this classification at random, but it is of great interest to know what regularities, if any, may exist in the post-organization classification.

Both of the questions discussed above are of interest in practical applications of pattern discriminators or recognizers, and in studies concerning the behavior of living organisms. It is hardly likely, for example, that exactly the same pattern repeats itself very often on exactly the same cells of a retina.

Again, considerable interest attaches in the study of animal behavior to phenomena of the following sort. If a rat is trained to discriminate by suitable behavior between solid vertical and horizontal rectangles, he will, without further training, discriminate by the same behavior between vertical and horizontal rows of dots. This action of classifying patterns which have not previously been seen is called perceptual generalization in psychology.[3] It should be noted that the rules which the generalization follows must be functions of the system under consideration, and could, in principle, be quite arbitrary. In general, for living organisms the rules presumably hold because they have survival value. Naturally the rules of most interest to us are those which we use ourselves. It would be most interesting if rules similar to some of ours could be demonstrated as properties of nonlinear element networks. It is also of interest to note that at least one well-known neurophysiologist feels that the basis of generalization lies in the nervous tissue itself.[4]

For these reasons, then, as well as for their importance in mechanical pattern classification, experiments were carried out to test whether the systems described above would continue to organize themselves subject to

[3] D. O. Hebb, "The Organization of Behavior," John Wiley & Sons, Inc., New York, N. Y., p. 12 ff.; 1949.

[4] K. S. Lashley, "The Problem of Cerebral Organization in Behavior," Vol. VII of Biological Symposia, Cattell, London, England, p. 302; 1942.

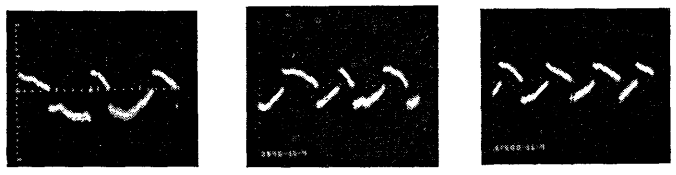
randomly varying inputs, and to examine what kind of classifications are made on the input set by nets trained with fixed and varying input patterns.

For these purposes, 16-element nets (8 input and 8 output) were used because it was desired to exhaust all possible input patterns, and we were limited to about 2^8 inputs by available time.

Results of First Experiment

In the first series of experiments "noisy" input patterns were formed from the patterns 11110000 and 00001111 by complementing the pattern digits at random at the instant of input to the net with a fixed probability ranging from about 0.06 to 0.25 in various experiments. This process can be considered as varying each pattern randomly about its mean of 11110000 or 00001111.

These two sets of "noisy" patterns were then used instead of p_1 and p_2 in the same type of experiment as described above. Fig. 3 shows the results of an experiment of this type run with complementation probability of 0.25. The first two trials were run without modifier for comparison. After the modifier was turned on, it can be seen that organization takes place as before. Rather more detailed fluctuation occurs during organization, which is of course due to pattern fluctuation. Such records were found to be typical among the half-dozen or so experiments of this type. Again, one or two failed to organize properly, at least during the time of observation, because of unfavorable special properties of the random initial connections. It is felt that with larger nets, the percentage of failures would be even smaller.



(a) (b) (c)

Fig. 3—Output record of 16-element net during organization. Input patterns p_1 and p_2 with 0.25 pattern digit complementation probability.

Results of Second Experiment

The second series of experiments was carried out to determine input pattern classification after organization with fixed patterns. Fixed inputs 11110000 and 00001111 were used to organize a given net. The modifier was then disabled so that no further changes in the net could occur and all 256 possible input patterns were then presented in turn. The output was set to zero immediately before each pattern was presented for a fixed test period. At the end of the test period, it was determined whether the magnitude of the output was less or greater than a fixed quantity, and, if greater, whether the output was positive or negative. This information serves to classify the input pattern into one of the three groups mentioned above. After recording the class, the next pattern in turn was tested.

Since organization of the net with p_1 and p_2 tends in a general way to connect each input subgroup with its corresponding output subgroup, a plausible type of input classification to be expected would be that input patterns with a preponderance of 1's in a given subgroup might be classified the same. That is, the patterns

1011 0000

1111 0101

1100 0000

might be grouped in the (+) class, whereas

0010 1110

0011 1110

1000 0101

might be grouped in the (−) class, and distributions with a balanced number of 1's would then be expected to fall in the (0) or neutral class.

To test this hypothesis a matrix of order 5 as in (1) was formed for each class according to its count of patterns having a given preponderance of 1's in a subgroup.

$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ 0 & & & \vdots & & \\ 1 & \cdots & & a_{21} & & \\ 2 & & & & & \\ 3 & & & & & \\ 4 & & & & & \end{array} \tag{1}$$

An element a_{mn} in this matrix then represents "a" patterns having "m" 1's in the first subgroup and "n" 1's in the second. The sum of all the a's in all three matrices must then total 256, the total number of input patterns.

For comparison, all possible patterns are represented in the single matrix:

$$\begin{array}{ccccc} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{array} \tag{2}$$

Thus, of the 256 patterns, 16 have 3 1's in the first subgroup, and a single 1 in the second, etc.

The three classification matrices of a typical net before organization are shown in (3).

$$\begin{array}{ccccc} \multicolumn{5}{c}{(+)} \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 3 & 5 & 0 \\ 0 & 3 & 7 & 7 & 2 \\ 1 & 2 & 5 & 10 & 4 \\ 0 & 0 & 2 & 4 & 1 \end{array} \quad \begin{array}{ccccc} \multicolumn{5}{c}{(-)} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 11 & 7 & 2 \\ 0 & 12 & 20 & 9 & 1 \\ 1 & 10 & 9 & 3 & 0 \\ 0 & 2 & 1 & 0 & 0 \end{array} \quad \begin{array}{ccccc} \multicolumn{5}{c}{(0)} \\ 1 & 4 & 6 & 2 & 1 \\ 4 & 16 & 10 & 4 & 2 \\ 6 & 9 & 9 & 8 & 3 \\ 2 & 4 & 10 & 3 & 0 \\ 1 & 2 & 3 & 0 & 0 \end{array} \tag{3}$$

Classification matrices of three nets after having been organized are shown in (4).

$$\begin{array}{ccc}
(+) & (-) & (0) \\
\begin{matrix} 0 & 0 & 2 & 0 & 0 \\ 0 & 8 & 10 & 0 & 0 \\ 1 & 20 & 16 & 2 & 0 \\ 2 & 14 & 14 & 3 & 0 \\ 1 & 4 & 4 & 0 & 0 \end{matrix} &
\begin{matrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 5 & 8 & 4 \\ 0 & 0 & 4 & 1 & 5 \\ 0 & 0 & 1 & 5 & 2 \\ 0 & 0 & 0 & 1 & 0 \end{matrix} &
\begin{matrix} 1 & 4 & 4 & 4 & 0 \\ 4 & 8 & 9 & 8 & 0 \\ 5 & 4 & 16 & 11 & 1 \\ 2 & 2 & 9 & 8 & 2 \\ 0 & 0 & 2 & 3 & 1 \end{matrix} \\
(+) & (-) & (0) \\
\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 5 & 1 & 0 \\ 1 & 7 & 14 & 4 & 0 \\ 2 & 9 & 12 & 2 & 0 \\ 1 & 4 & 3 & 0 & 0 \end{matrix} &
\begin{matrix} 0 & 0 & 3 & 3 & 1 \\ 0 & 0 & 11 & 10 & 2 \\ 0 & 0 & 16 & 8 & 1 \\ 0 & 0 & 8 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{matrix} &
\begin{matrix} 1 & 4 & 3 & 1 & 0 \\ 4 & 14 & 8 & 5 & 2 \\ 5 & 17 & 6 & 12 & 5 \\ 2 & 7 & 4 & 12 & 4 \\ 0 & 0 & 2 & 4 & 1 \end{matrix} \\
(+) & (-) & (0) \\
\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 9 & 3 & 0 \\ 1 & 14 & 23 & 6 & 0 \\ 3 & 13 & 16 & 3 & 0 \\ 1 & 4 & 3 & 1 & 0 \end{matrix} &
\begin{matrix} 0 & 0 & 3 & 2 & 1 \\ 0 & 0 & 7 & 6 & 3 \\ 0 & 2 & 7 & 8 & 6 \\ 0 & 1 & 3 & 6 & 4 \\ 0 & 0 & 1 & 2 & 1 \end{matrix} &
\begin{matrix} 1 & 4 & 3 & 2 & 0 \\ 4 & 12 & 8 & 7 & 1 \\ 5 & 8 & 6 & 10 & 0 \\ 1 & 2 & 5 & 7 & 0 \\ 0 & 0 & 2 & 1 & 0 \end{matrix}
\end{array} \tag{4}$$

The last net was operated with considerably more noise than usual added into the excitation at the element thresholds, as an additional experiment.

Upon inspection of the above matrices, it can be seen that the effect of the organization is to cause the entries in a (+) matrix to tend to cluster in the lower left corner, while those in a (−) matrix tend to cluster in the upper right corner. These tendencies are to be expected on the basis of our hypothesis, and perfect classification would make just three classes, namely lower left, upper right corner, and diagonal, corresponding to (+), (−), and (0), respectively. Of course, it might be expected that patterns next to the diagonal may be misclassified, inasmuch as they correspond to a preponderance of only a single 1.

Even in those cases where gross misclassifications occur, it will generally be found that they are a minority of the total possible cases in that cell. However, there are a few exceptions such as the 10 cases in the matrix cell 1, 2 in the first (+) matrix of (4).

When several of the patterns in a cell m, n are misclassified by two or more nets, it would not be expected that these would all be the same pattern, so that if all the nets were consulted for each pattern and majority rule accepted, the results should be improved. This plan was tried for the three nets above. The resulting matrices are shown in (5).

$$\begin{array}{ccc}
(+) & (-) & (0) \\
\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 6 & 0 & 0 \\ 1 & 14 & 18 & 0 & 0 \\ 2 & 14 & 13 & 0 & 0 \\ 1 & 4 & 3 & 0 & 0 \end{matrix} &
\begin{matrix} 0 & 0 & 2 & 2 & 1 \\ 0 & 0 & 8 & 9 & 4 \\ 0 & 0 & 7 & 7 & 5 \\ 0 & 0 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 \end{matrix} &
\begin{matrix} 1 & 4 & 3 & 2 & 0 \\ 4 & 12 & 7 & 4 & 0 \\ 6 & 10 & 7 & 8 & 1 \\ 2 & 3 & 4 & 9 & 2 \\ 0 & 0 & 1 & 4 & 1 \end{matrix}
\end{array} \tag{5}$$

Thirty-one patterns of the 256 were classified differently by each of the three nets, and are not included.

A definite improvement in classification can be seen in (5), especially in the (+) and (−) classes. A large fraction of possible cases are classified according to a preponderance of 1's in most cases. There are still quite a few (0) cases mixed up, but in many of these there is a preponderance of 1's of only one or two. This kind of error is probably accentuated in nets as small as these, because a maximum preponderance of only four is available. It would be expected that in larger nets the results would be improved, since they would have much greater preponderant differences, and should supply better statistics generally.

It should be noted also that no attempt was made to optimize the classification criteria themselves.

Classifications after organizing with "noisy" input patterns were also tried, with much the same results as those described above.

Discussion of Results

We have seen that the system under discussion generalizes from p_1 vs p_2 discrimination to a "1's preponderance" classification. The generalization is statistical in nature, and a number of nets in parallel with majority decision may be needed for more nearly perfect performance.

It is of interest to discuss the applications such a generalization may have to pattern recognition.

We will assume that the principle can be made to apply to systems with many inputs, as there is no known obstacle to using larger nets, or combinations of smaller ones, to achieve this result.

In the first place, generalizations of the horizontal-vertical kind mentioned earlier which are made by the rat, can be effected. Suppose that the whole picture field is made up of a mosaic of input elements, like a retina. Suppose also that the field itself is always mapped ontp the same elements, i.e., that some kind of centering scheme is used. Then let the system be trained to distinguish between horizontal and vertical rectangles, as shown in Figs. 4a and 4a′, each rectangle being thought of as made up of a large group of input elements such as we have been considering. There will be an overlap of elements at the center, but total overlap elements are a small percentage of the total in either group and should present no difficulty to acquisition of the discrimination. Now, if preponderance generalization has taken place, discrimination will still take place "correctly" as between vertical and horizontal rows of dots, or other horizontal-vertical figures. In fact, as long as a preponderance of inputs of a test figure lies within the set of inputs of one of the original training patterns, the system will classify them alike. If some of the test figure inputs lie outside both the original patterns, no difficulty would be anticipated, since these inputs should remain neutral or at least inactive. Fig. 4 shows some of the cases in which successful horizontal-vertical discrimination would be expected.

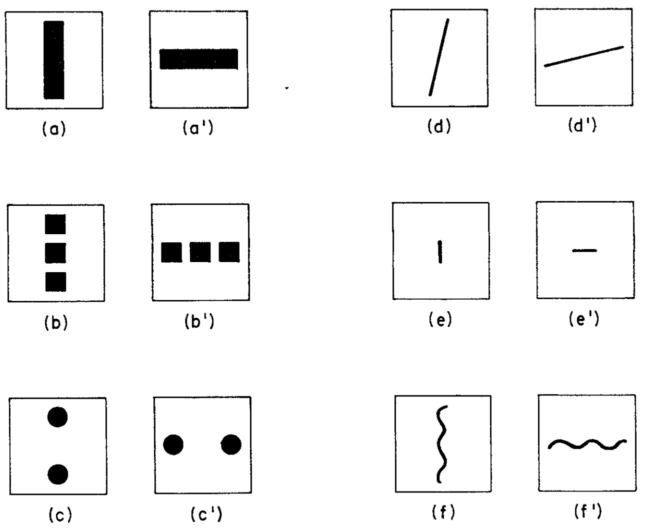


Fig. 4—Horizontal and vertical discrimination patterns.

There is still another possibility in generalization, if the input patterns contained fluctuation, either inherent or artificially introduced. We have seen that discrimination of such patterns can be organized by our system. During such training a "haze" of patterns would be expected to form about the "learned" pattern which will determine an additional set of patterns producing a like response, and hence being classified the same. In this way, a figure will still be classified correctly after having been translated or rotated slightly, for example. This is another instance in which a rat may be used as an example, since he will still recognize a triangle after it has been rotated a few degrees.[5]

These examples serve to show some of the features of pattern recognition which may be effected by a primitive generalization like that found to be a property of the self-organizing nets described. Comparisons to recognition behavior of a rat which have been made are not, of course, meant to imply that it is thought that the same mechanism is in operation, since there are other generalizations shown by the rat not possessed by our nets. However, it is clear that crude but useful generalization properties are possessed even by randomly connected nets of the type described.

Notes on the Simulation Program

The simulation program proceeds from a description of the class of networks under investigation and generates members of the class by selection of particular values of the description parameters at random. These parameters include quiescent thresholds, refractory periods, firing delay periods, input and output group size, threshold and excitation decay constants, and certain parameters which prescribe the manner in which elements are to be interconnected. Element i is connected to element j with a probability P_{ij} which can be made to depend on i and j and on any characteristics of the network as a whole. In the class of networks we have studied, P_{ij} is constant and equal to 0.75, i.e., the networks all have a constant mean connectivity of 0.75.

In carrying out the simulated activity of a net, the time parameter is held fixed while the program scans through a list of the elements calculating current values of threshold and excitation, and carries out the firing cycle changes as required. The time parameter is then advanced one unit and the process is repeated.

For a net of 16 elements, the real time consumed during one simulated time unit is about one-tenth of a second, varying according to the amount of activity in the net. The training of a typical net takes place in about 5 minutes. The subsequent classification of the complete set of 256 input patterns requires about 45 minutes.

The simulation program required about 1,000 16-bit registers of storage. Utility programs such as printouts, displays, and punched tape routines occupied another 1,000 registers. The remainder of the 4,096 registers of the MTC memory was divided between storage of the network structure itself and summary tables for classified patterns.

It may be of interest to note some of the computer design features which are desirable in simulation work of this kind. Most important, of course, is a large random-access memory of high speed. While this is generally taken as a basic requirement of any modern general-purpose computer, it is particularly important in programs which deal with the interaction of many elements with one another. In general, it would be more useful to have this memory in the form of many registers of small word-length rather than few registers of large word-length. In lieu of a "broad, shallow" memory, special instructions which permit the efficient extraction and replacement of parts of a stored word would be very valuable.

In many respects a simulation program of this type is like a large problem in bookkeeping and as such would make efficient use of design features which facilitate the repetition of operations on successive sets of data. For example, an arithmetic element capable of performing parallel operations on several quantities simultaneously would materially reduce the computing time. Similarly, "index registers" which modify the address part of indicated instructions in the program, so that the instructions refer automatically to successive sets of data, might well increase the effective computing speed by a factor of two in this application.

The requirement of randomness is generally met by the use of "pseudo-random" number sequences calcu-

[5] K. S. Lashley, "In Search of the Engram," No. IV of Symposia Soc. for Experimental Biology," Physiological Mechanisms in Animal Behavior," Academic Press, New York, N. Y., p. 473; 1950.

lated by the program. In the event that the simulation experiment makes extensive use of such randomness it would be desirable to incorporate a source of uniformly distributed random numbers as one of the electronic elements of the computer. Such an element would, of course, also be of great value in statistical work and monte-carlo calculations in general.

Finally, it is worth mentioning that simulation experiments involving partially random program behavior, unlike arithmetic computations, generally require the presence of the experimenter at the computer, at least during the program checkout phase and subsequently whenever large changes in operating parameters are made. For this reason any features which assist the experimenter in evaluating the operation of various parts of the program "on the spot" are of great value. In this category one might include programmed cathode-ray tube displays, audio output, and the ability to print out selected memory registers without stopping the computer.

Pattern Recognition and Modern Computers*

O. G. SELFRIDGE†

Introduction

WE CONSIDER the process we call Pattern Recognition. By this we mean the extraction of the significant features of data from a background of irrelevant detail. What we are interested in is simulating this process on digital computers. We give examples on three levels of complexity corresponding to the subjects of the other three speakers here today. We examine in detail the problem on the second level, visual recognition of simple shapes.

Finally, we show how our attack on that problem can be extended so that the computer is essentially performing a learning process and constructing new concepts on the basis of its experience.

Pattern Recognition

By pattern recognition we mean the extraction of the significant features from a background of irrelevant detail. We are interested in simulating this on digital computers for several reasons. First, it is the kind of thing that brains seem to do very well. Secondly, it is the kind of thing that computing machines do not do very well yet. Thirdly, it is a productive problem—it leads naturally to studying other processes, such as learning. And, finally, it has many fascinating applications on its own.

We shall not review here the valuable work that has been done and is being done elsewhere.

Examples of Pattern Recognition

Consider Fig. 1. The horizontal lines on the left differ from those on the right in having vertical spikes mostly at the left end. That is, here there are two patterns: those with a preponderance at the left end and those with a preponderance at the right end. The notion of simple preponderance or elemental discrimination is clearly one of the most primitive sources of patterns.

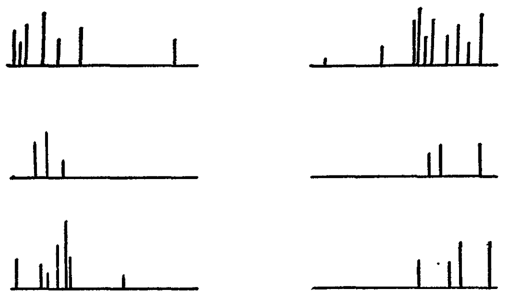


Fig. 1

Here we have filtered each line from perhaps 100 bits down to just one. It is this filtering that is pattern recognition.

Our next example is the visual recognition of simple shapes. This is a two-dimensional problem, of course, while the previous one was merely one-dimensional. Both the shapes in Fig. 2 are clearly squares though (1) they are in different places, (2) they have different sizes, (3) one is hollow, the other not, and (4) they have different orientations.

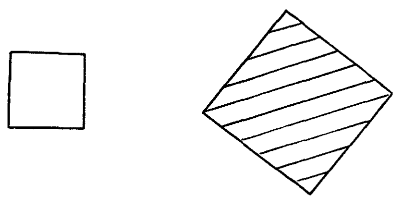


Fig. 2

Our final example, like our first, divides all the configurations of data into two classes. From every chess

* The work in this paper was sponsored jointly by the U. S. Army, U. S. Navy, and U. S. Air Force under contract with M.I.T.

† Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, Mass.

position, that is, from every arrangement of chess pieces on a chess board, it either is or is not possible for White to force a win. Those from which White *can* force a win, we call "good" positions and the rest we call "bad." It is clear that recognition of this kind is a good deal more complicated than in either of the other examples. In itself, it is obviously equivalent to the whole game. For any dunce, playing White, could inevitably win by playing only moves which led to positions which he could recognize as "good."

Significance: The Criterion

As we have seen, pattern recognition involves classifying configurations of data into classes of equivalent significance so that very many different configurations all belong in the same equivalence class. I repeat our definition: Pattern recognition is the extraction of the *significant* features from a background of irrelevant detail.

Probably the most important word here is "significant." Now significance is a function of, first, context, and second, experience. As in Fig. 3, the shape is recognized with the help of its context. Now, of course, context is a function of experience. But more than that, experience alone affects the *kind* of thing we regard as significant.

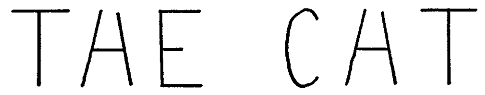


Fig. 3

In this way, the whole process of Pattern Recognition is inevitably tied up with ways of determining significance. I suggest—this is my own fancy—that this is the distinction usually made between machines and men. That men can learn by experience to extract and deal with the *significant* things and machines cannot . . . I do not, however, believe it is a valid distinction.

The other three speakers will talk in detail of pattern recognition on the three levels I have introduced to you. However, I should now like to examine the model used by the next speaker. His problem is the recognition of simple shapes in a visual field, such as block capital letters.

Letter Recognition

The problem is this: We want the computer to recognize block capital letters, however they are drawn. Let us list the variations that ought not affect the valuation:

1. Over-all size, between wide limits.
2. Position.
3. Orientation, within limits.
4. Angular separation of the two uprights.
5. Relative lengths of the two uprights.
6. Thickness, and changes in thickness, of the line segments, within limits.

and so on.

It is clear, I think, that if we tried to construct a template for every *A*, we should have an embarrassingly large number of them. *Some* filtering of the image has to be done in some way.

Now, people surely use many kinds of features to recognize things. The angularity and size of a letter are instances and they are not related in any obvious way. The features that our model extracts are those derivative from a few simple local image transformations.

Fig. 4 shows a block diagram of our model. The *original image* is transformed into a *secondary image* by operation *A*, *B*, or *C*. The secondary image itself may be transformed a number of times. Then, after the image has been transformed by a *sequence* of operations, the number of blobs remaining is counted by the black box called the Blob Counter. A typical original image is transformed sequentially in Fig. 5, showing the secondary images at each step.

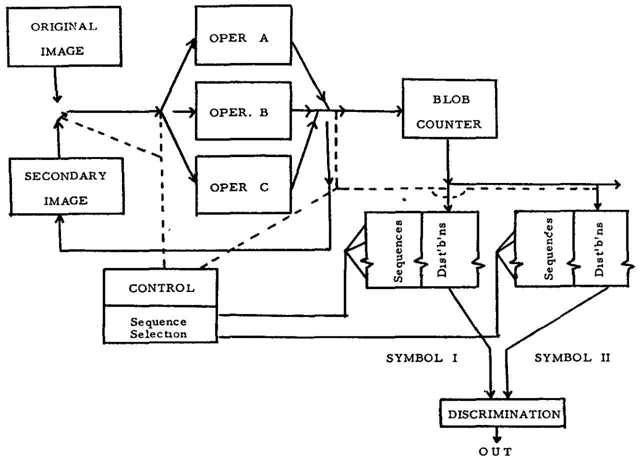


Fig. 4

The number coming out of the blob counter is then compared with the numbers stored for that particular sequence under the various symbols. If, after a number of sequences have been run on an image, the numbers check sufficiently well with the stored distribution of Symbol *I*, say, the computer may identify the image as Symbol *I*.

When we were constructing our transformations, we already had in mind some of the tasks they ought to do. First, we wanted to be able to eliminate small pieces of noise, isolated 1's in a field of 0's, say. The averaging operation, our first operation, does approximately this job. This operation emphasizes local homogeneity.

Applying this transformation to an image produces a new image where the intensity at each point is a kind of average of the intensities of the points surrounding it in the original image. Thus solitary 1's in a field of 0's (or vice versa) may be eliminated.

Our second operation emphasizes local discontinuities and tends to bring out edges and corners.

It does this by evaluating the radial asymmetry around each 1 in the image, and keeping only those 1's which have more than a certain amount. That is, our second operation emphasizes local inhomogeneity.

Other operations might well be variations of these, or perhaps something very different. Dr. Dinneen, the next speaker, will discuss the details of programming these operations and the results. He will also go into some extensions, and what kind of things we èxpect the operations to be able to do.

One of the things we know that they can do is to extract corners; for example, see Fig. 5. After clearing the image of noise with the averaging operation A, and edging twice with operation B, we have four blobs at the four corners of the original square. Now, clearly, having four corners is a significant criterion for squares: not that anything with four corners is a square, but no triangle has four, and no square has three. This sequence can, therefore, successfully distinguish squares from triangles.

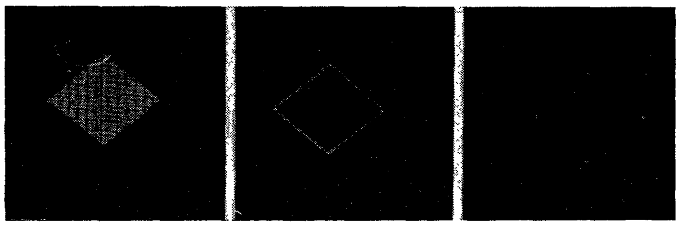


Fig. 5

Note that many kinds of features cannot be handled by our particular model, which only counts certain kinds of features. For example, our model could never distinguish a C from a U; the latter might be merely the former rotated through 90 degrees, and all the operations that we have considered so far are unpolarized, that is, do not distinguish much between different directions.

Eventually, we hope to be able to recognize other kinds of features, such as curvature, juxtaposition of singular points (that is, their relative bearings and distances), and so forth.

I shall now discuss our plans for having the computer itself hunt for good sequences and assign the proper values. Every sequence is good or bad according as the numbers obtained from applying it to images tend to differ consistently for different symbols.

Now the search for "good" sequences—that is, those that can discriminate between symbols—can, of course, be programmed to be entirely random. For example, one could start with two sequences and always keep the better one, discarding the old, and picking a new sequence completely at random. But it seemed to us that this was not a realistic way for the computer to improve its performance.

Rather we shall try to have the computer do what I feel we do in such a case. It must build new sequences *like* the sequences that have worked well before. It must begin with sequences chosen at random, because it must have no *a priori* knowledge of the things it sees.

After the sequences have been tested for some time, the computer will have built up a distribution function for each sequence and each symbol.

We give our computer a few programmed operations, A, B, and C, and a random number generator; let it then pick out randomly a few short sequences of operations. It has no way of knowing yet whether these sequences yield any significance. We now feed the machine A's and O's, telling the machine each time which letter it is. Beside each sequence under the two letters, the machine builds up distribution functions from the results of applying the sequences to the image. Now, since the sequences were chosen completely randomly, it may well be that most of the sequences have very flat distribution functions; that is, they have no information, and the sequences are therefore not significant. Let it discard these and pick some others. Sooner or later, however, some sequences will prove significant; that is, their distribution functions will peak up somewhere. What the machine does now is to build up new sequences *like* the significant ones. This is the important point. If it merely chose sequences at random it might take a very long while indeed to find the best sequences. But with some successful sequences, or partly successful ones, to guide it, we hope that the process will be much quicker. The crucial question remains: how do we build up sequences "like" other sequences, but not identical? As of now we think we shall merely build sequences from the transition frequencies of the significant sequences. We shall build up a matrix of transition frequencies from the significant ones, and use these as transition probabilities with which to choose new sequences.

We do not claim that this method is necessarily a very good way of choosing sequences—only that it should do better than not using at all the knowledge of what kind of sequences has worked. It has seemed to us that this is the crucial point of learning.

In general, if one wants a computer to do some job, one must merely specify sufficiently precisely what that job is. In some way the enormous amount of information in experience must be filtered down to the significant or valuable part. This process, which we call pattern recognition, may be more or less complex, as the other speakers illustrate. What we have tried to do is to specify one way in which certain simple visual patterns can be recognized by the computer, and in which the computer may improve its recognition by learning.

Programming Pattern Recognition*

G. P. DINNEEN†

EVERYONE LIKES to speculate, and recently there has been a lot of talk about reading machines and hearing machines. We know it is possible to simulate speech. This raises lots of interesting questions such as: If the machines can speak, will they squawk when you ask them to divide by zero? And can two machines carry on an intelligent conversation, say in Gaelic? And, of course, there is the expression "electronic brain" and the question, Do machines think? These questions are more philosophical than technical and I am going to duck them.

Over the past months in a series of after-hour and luncheon meetings, a group of us at the laboratory have speculated on problems in this area. Our feeling, pretty much unanimously, was that there is a real *need* to get practical, to pick a real live problem and go after it. As the title of this session suggests, we picked pattern recognition. Selfridge has already explored some of the implications of this problem and indicated at least three levels of complexity.

Selfridge and I have begun an investigation of one aspect of pattern recognition, that is, the recognition of simple visual patterns. Consider, for example, how many different representations of the block capital A we recognize as A. A great number of variations in such things as orientation, thickness, and size can occur without loss of identity. Our real live problem then is to design a machine which will recognize patterns. We shall call any possible visual image, no matter how cluttered or indistinct, a configuration. A pattern is an equivalence class consisting of all those configurations which cause the same output in the machine. For example, if the machine had just two outputs, yes and no, then one pattern would be all those inputs which caused the machine to say no.

The visual patterns we have used in our experiments have been mostly block capital A's and O's, and some squares and triangles. Why did we choose letters? Probably because of the interest in reading machines. Why A and O? Perhaps because O upside down is still O, like radar spelled backwards. Capital letters and geometric figures seem like a good start. Straight lines are too simple and most other things, such as faces, are too complicated. Let me emphasize again that we are not trying to build an efficient reading machine, but rather to study how such a machine might work.

* The research in this document was supported jointly by the Army, Navy, and Air Force under contract with the Massachusetts Institute of Technology.

† Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, Mass.

Our theory of pattern recognition is that it is possible to reduce by means of a sequence of simple operations a configuration to a single number, or by means of a set of such sequences to a set of numbers. We believe that for the proper sequences, almost all of the configurations belonging to a given pattern reduce to the same number or set of numbers. At least one of the operations of this machine must be a counting operation.

A machine for pattern recognition should place a configuration, such as one representation of the block capital A, into its proper equivalence class. The decision as to what things belong to an equivalence class or pattern is made at this stage by the designer. The machine must extract the essential characteristics of the configuration in order to recognize the pattern to which it belongs.

One of the first steps is the design of basic operations which the machine uses to reduce the input image. Although this is a preliminary and experimental phase of the study, the results obtained have been very interesting and extremely valuable in formulating future studies. For this reason, it seems worthwhile to discuss these results now.

The problem we have stated is not basically arithmetic, and it is not even clear that a machine for pattern recognition should be completely digital. However, one of the very important applications of a high-speed digital computer is the *simulation* of just such problems. We have used MTC—the Memory Test Computer—at Lincoln Laboratory for our investigation.

The Image

From now on, we are considering the simulation of this problem on MTC. The first problem is "how does MTC see the image?" The field of view is a 90×90 array (Fig. 1). A picture is constructed by making each of the 8,100 cells black or white, similar to a newspaper photograph, except that there are only two gradations in intensity, zero to one. If black is interpreted as one and white as zero, and storage registers are assigned to the 90×90 array, six to a row, as in Fig. 2, a simple image may be described by giving the contents of the 540 register as octal numbers. A flexowriter tape giving the contents of 540 registers is read into the machine and the visual image is thus stored by the computer. We chose a 90×90 array because we wanted the image to be as large as was practical in terms of computer storage. The larger the image, of course, the greater the resolution and the variety of inputs.

For purposes of pattern recognition, certain basic operations will be performed on the input image. Each operation performs a mapping of the input image, or working image. The result of this mapping is also stored in the machine. Thus another bank of 540 registers is required for storage of this new image. If a sequence of operations is to be performed, then the transformed image is transferred back to the input storage or working image after each operation. If it is desired to retain the original image, then three banks are required, as shown in Fig. 3.

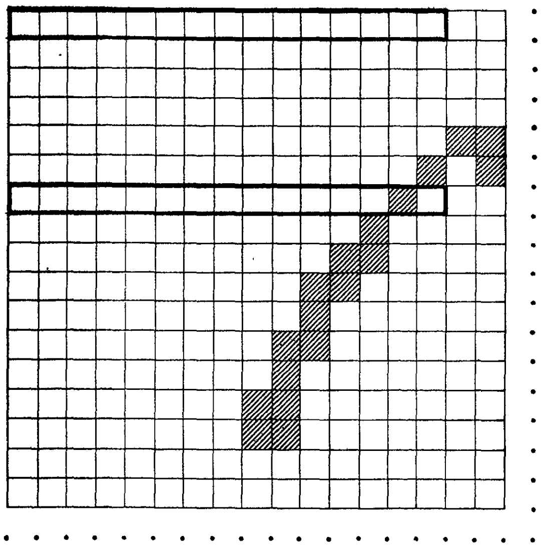


Fig. 1—90×90 input array.

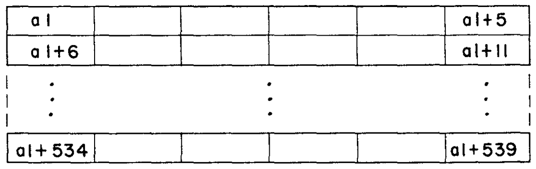


Fig. 2—The input image.

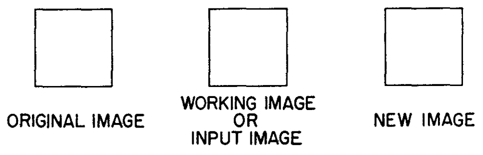


Fig. 3

Averaging Operation

The first of the basic operations is an averaging operation which was designed to eliminate spurious noise and to smooth irregularities in the image. The point of this operation is that those elements or cells which agree with their neighbors are unchanged, but those which do not are changed. The notion is that the resulting image is more homogeneous. The averaging operation is performed by observing the contents of all the elements in some $n \times n$ square surrounding an element. In Fig. 4 we are considering element a_{ij}, thus we look at the elements in the 5×5 square surrounding it. Let us agree to call this square a "window." To complete the operation we count the number of ones in this window and compare the total with some threshold. If the count is greater than or equal to the threshold then the corresponding element in the new image, a_{ij}' is "one," otherwise $a_{ij}' = 0$. For example, a zero surrounded by all zeros remains zero, but a zero surrounded by some ones may become zero or one, depending on the threshold.

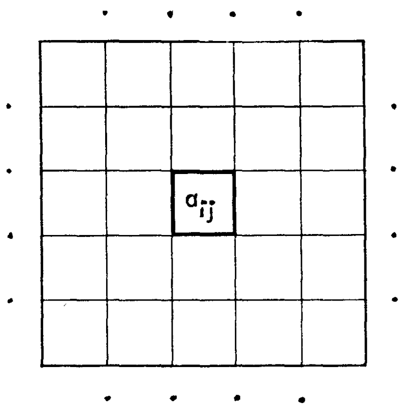


Fig. 4—Averaging operation.

The first averaging operation uses a 5×5 square. A count of the number of ones inside this square is made (Fig. 4).

$$N = \sum_{m=-2}^{+2} \sum_{n=-2}^{+2} a_{i+m,i+n}.$$

The N is compared with the threshold T and

$$\begin{aligned} N &\geqq T \qquad & a_{ij}' &= 1 \\ N &< T \qquad & a_{ij}' &= 0. \end{aligned}$$

The threshold T is held constant over the entire working image.

Results of the Averaging Operation

A large number of experimental runs were made using the computer to test the operation. As predicted, averaging with a 5×5 window and a threshold of 5 eliminated scattered "ones" and filled in holes. Fig. 5 is an input image, and Fig. 6 is the same image after this averaging operation. These are photographs of one of the MTC scopes.

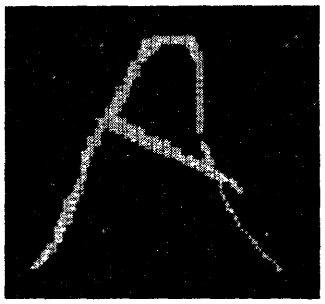


Fig. 5—A3, input image.

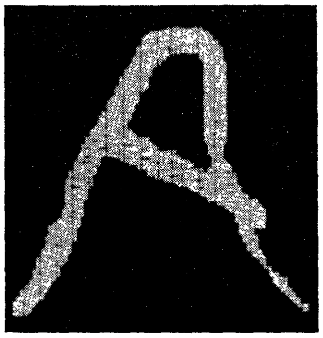


Fig. 6—A3, after averaging with threshold 5.

For a low threshold, such as 5 for a 5×5 window, the image will be thickened. As the threshold is raised a thinning takes place. This is evident in Figs. 7 through 11. It is particularly significant that for a threshold of 15, the corner point and two junction points are isolated. The same phenomenon is shown in Figs. 12 through 17. The thick A of Fig. 18 has a blank strip and one small hole. For the low thresholds these irregularities are removed and for the high thresholds they are emphasized, as shown in Figs. 18 through 25.

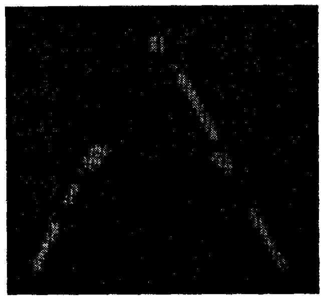


Fig. 10—A4, after averaging with threshold 13.

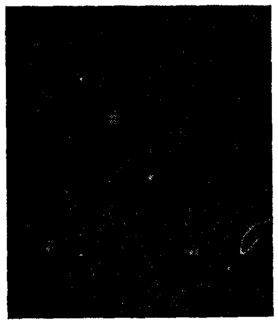


Fig. 11—A4, after averaging with threshold 15.

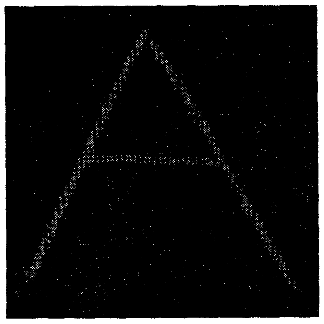


Fig. 7—A4, input image.

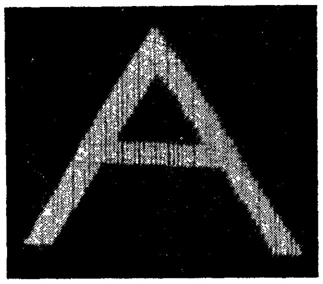


Fig. 12—A5, input image.

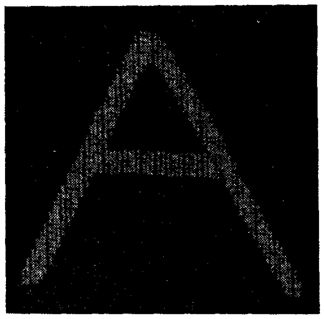


Fig. 8—A4, after averaging with threshold 5.

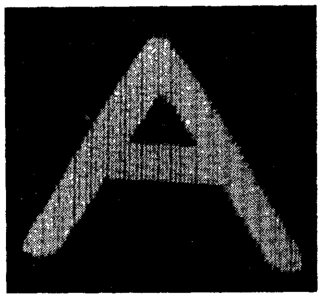


Fig. 13—A5, after averaging with threshold 5.

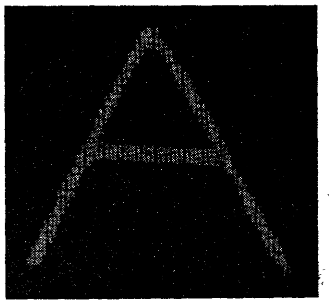


Fig. 9—A4, after averaging with threshold 10.

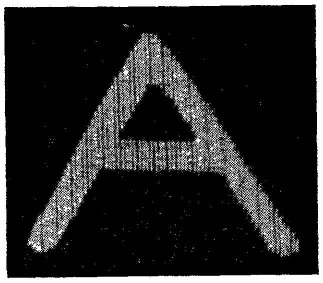


Fig. 14—A5, after averaging with threshold 10.

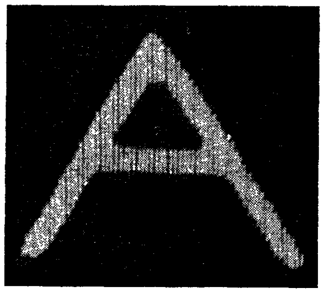


Fig. 15—A5, after averaging with threshold 15

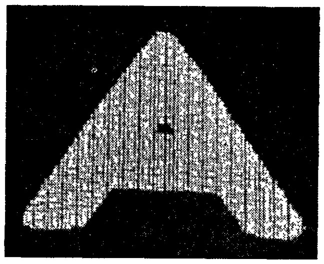


Fig. 20—A6, after averaging with threshold 5.

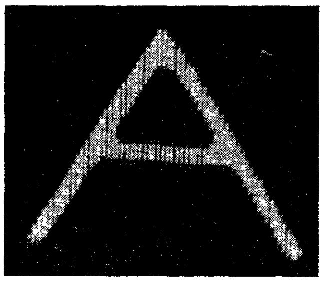


Fig. 16—A5, after averaging with threshold 20.

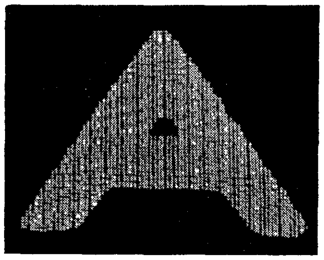


Fig. 21—A6, after averaging with threshold 10.

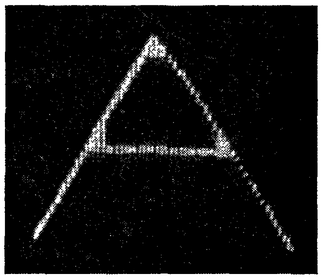


Fig. 17—A5, after averaging with threshold 25.

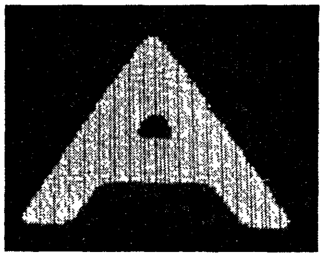


Fig. 22—A6, after averaging with threshold 15.

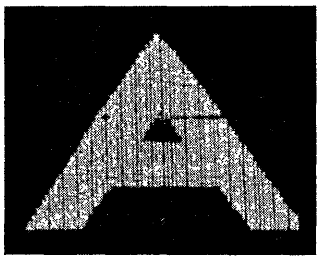


Fig. 18—A6, input image.

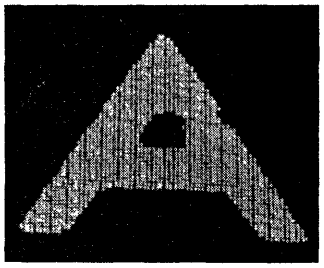


Fig. 23—A6, after averaging with threshold 20.

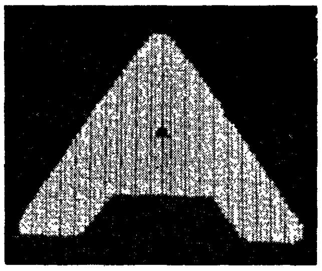


Fig. 19—A6, after averaging with threshold 3.

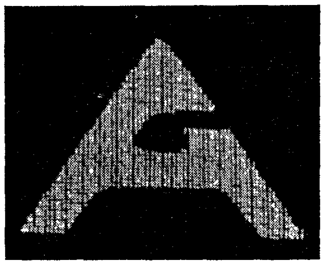


Fig. 24—A6, after averaging with threshold 22.

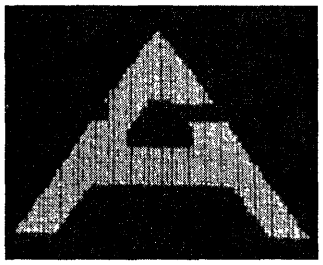


Fig. 25—A6, after averaging with threshold 25.

We see that for rough smoothing we like a low threshold, and for thinning we like a high threshold. Our notion of sequence of operations is now brought to mind. For if a letter such as the last one were first averaged with a low threshold the gaps would be filled. Then successive averaging with a higher threshold would thin the letter. The problems one encounters in this next phase are apparent. How often should these operations be repeated? What thresholds should be used?

You will have noticed that although the transformation is not sensitive to small changes in the threshold, large changes have a major effect. To attack the problem of setting the threshold, we have considered another averaging operation where the threshold for the small 5×5 window is determined by the degree of homogenity inside a larger window, say 15×15. In particular, for each cell we count the number of ones inside a 15×15 window surrounding it and use this number to set the threshold for the 5×5 window. This is like an automatic gain control. When the 15×15 window is very dense, we use a high threshold, and conversely. This operation has been tried and the results look favorable, but unfortunately we have no photographs yet.

An Edging Operation

The edging operation, quite unlike the averaging operation, preserves elements which are centers of asymmetry, that is, those which are located in the regions of discontinuity of the image. The operation sharpens differences. It picks out the edges of letters and for certain choices of thresholds locates the corners, junctions, and end points of letters. This operation is performed by observing the contents of all the elements in some $n \times n$ square surrounding an element which is one; zeros remain zero. In Fig. 26 we are considering element a_{ij}.

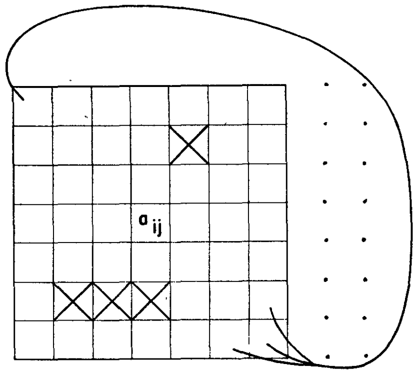


Fig. 26—The edging operation.

The edging operation is like a two-dimensional derivative, since we count changes about the center element. In particular, we scan the elements in the 7×7 window starting at $a_{i-3,j-3}$. When we reach a "one" we observe the three diagonally opposite elements in the same ring. If these are all zero we count one, otherwise we count zero. We consider the three diagonally opposite bins so that lines with small curvature give a zero count of differences. After the entire window has been scanned, with the exception of element a_{ij}, the total count of differences is compared with a threshold T. If the count is greater than the threshold, we enter a one in the element a_{ij}' of the new image. If the count is less than or equal to the threshold, we enter a "zero" in the element a_{ij}' of the new image. To fix the threshold, we first count the total number of ones in the window and take some proportion of this as the threshold. A further modification of this operation is the weighting of the count of differences depending on the ring, that is, we put weight W_1 on the elements in the first ring around the element a_{ij}, W_2 for the second ring, and so on. To identify this operation, we must specify the size of the window, the weighting, and the threshold.

We first tried an edging operation which used a 7×7 window with a weight of two on all differences which occur in the first and second ring, and a weight of one on all differences in the outer ring. The total number of ones in the 7×7 window, exclusive of the center element, will be denoted by N. The threshold T of the operation was taken to be $(j/8) \cdot N$ for $j = 0, 1, 2, \cdots, 8$.

Results of the Edging Operation

In testing the edging operation we were interested in its ability to pick out singular points, that is, junction points, end points, corners in the case of letters, and in its ability to transform the input image into a new image where only the outline remains.

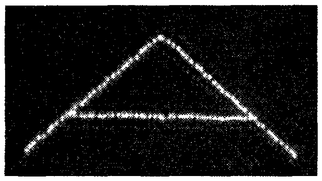


Fig. 27—Thin A.

For an input image, Fig. 27, edging with thresholds 2/8, 4/8, and 6/8 of the ring total isolates the singular points as shown in Figs. 28, 29, and 30. Edging with a threshold of 0/8 of the ring total is equivalent to removing only those points which are centers of symmetry. This effect is shown in Fig. 31 for the input image of Fig. 12. The outline of the letter is shown in Fig. 32, the result of edging with a threshold of 2/8 times the ring total. For a thick *A* (see Fig. 18) the results are shown in Figs. 33 through 35. It is significant that the irregularities are emphasized for low thresholds and suppressed for the high thresholds. Figs. 36 through 39 demonstrate the way corners are isolated in a square and the edging effect for a 0 is shown in Figs. 39 and 40.

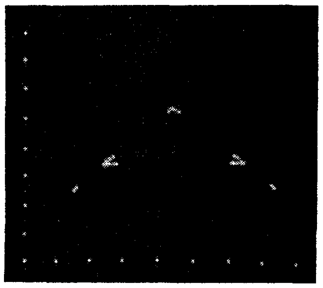


Fig. 28—Edging with threshold 2/8.

Fig. 29—Edging with threshold 4/8.

Fig. 30—Edging with threshold 6/8.

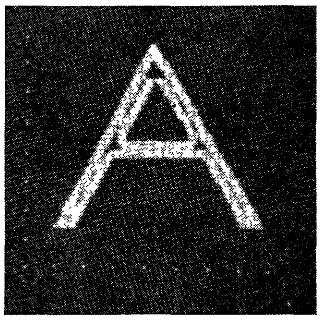


Fig. 31—Edging with threshold 0/8.

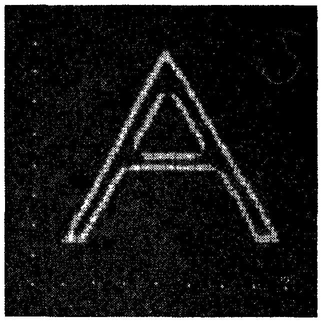


Fig. 32—Edging with threshold 2/8.

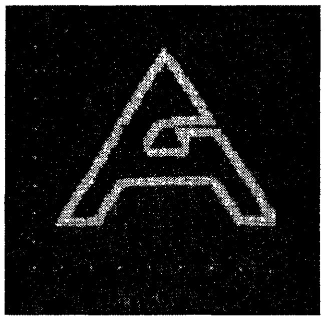


Fig. 33—Edging with threshold 0/8.

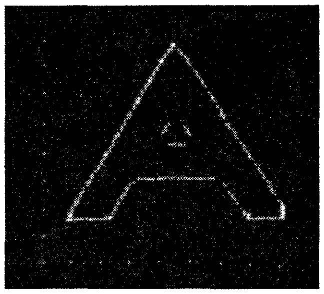


Fig. 34—Edging with threshold 4/8.

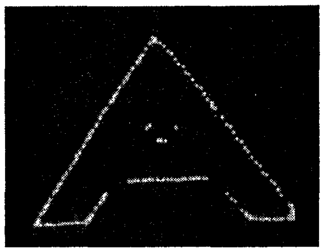


Fig. 35—Edging with threshold 6/8.

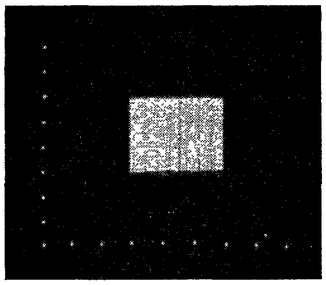


Fig. 36—Square.

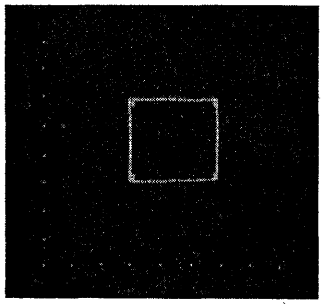


Fig. 37—Edging with threshold 4/8.

Fig. 38—Edging with threshold 8/8.

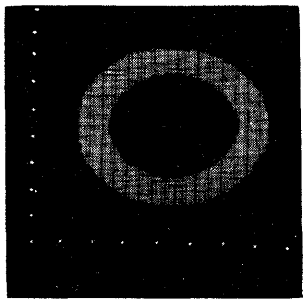


Fig. 39—Thick 0.

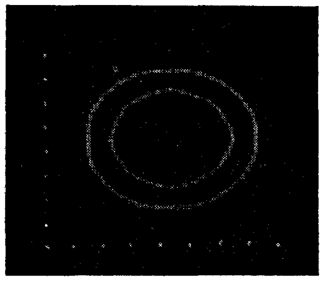


Fig. 40—Edging with threshold 2/8.

Other Operations

Before I say anything about new operations, a word about our reasons for these first two operations is in order. We tried to pick operations which were simple in structure and unrelated to particular patterns. In other words, we avoided special operations which might work very well for A's and O's, but not very well for other shapes. There is evidence in neurophysiology, moreover, that both averaging and edging of visual images are performed by the nervous system of many animals.

Many other operations have been proposed but I would like to discuss just one of these. Selfridge talked about identifying a letter by counting blobs. By blob we mean a region of ones with no separation of more than one cell. For example, there were five blobs in Fig. 29. A blob counter is then simply an operation which counts these blobs. We propose to do this by scanning the image until a one is reached. Then a rectangle with a perimeter two cells wide will be constructed and gradually enlarged until the entire perimeter is empty. Thus the blob is isolated and counted.

Certainly some operation to measure curvature will be needed, as well as some way to describe relative positions of blobs.

Although the first two operations are very simple, the coding of the program is quite complicated. This is due to a large extent to the necessity of using the individual bits of the storage registers as independent storage bins. We need a lot of shifting and cycling; the simple averaging operation takes about 300 instructions and requires about 20 seconds a letter. The two window averaging takes about 400 instructions and requires about 4 minutes. The edging requires 700 registers of instructions and takes 2 minutes.

Conclusion

In his paper, Selfridge has given our model for pattern recognition. I think we have shown that even with just two simple operations, a sequence can be found such that some A's can be reduced to five blobs. One of our next tasks will be to choose such a sequence, try it on many A's, and plot the distribution of the number of blobs. In this way we shall determine the useful sequences. This leads naturally to the next phase of the study, which is the study of the model for pattern recognition.

Acknowledgment

For their advice and encouragement, I wish to thank J. V. Harrington, I. S. Reed, O. G. Selfridge, H. Sherman, and G. E. Valley, who participated in the seminar which launched this study. It would have been impossible to obtain the experimental results without the excellent assistance of Miss B. Jensen and Mrs. M. Kannel in programming, coding, and operating the Memory Test Computer. I wish to acknowledge the excellent co-operation of those responsible for the operation of the Memory Test Computer.

The Chess Machine: An Example of Dealing with a Complex Task by Adaptation

ALLEN NEWELL†

THE MODERN general-purpose computer can be characterized as the embodiment of a three-point philosophy: (1) There shall exist a way of computing anything computable; (2) The computer shall be so fast that it does not matter how complicated the way is; and (3) Man shall be so intelligent that he will be able to discern the way and instruct the computer.

Sufficient experience with the large machines has accumulated to reveal the peculiar difficulties associated with these points. There has been a growing concern over problems which violate them, and instead satisfy the condition that (1) The relevant information is inexhaustible; (2) The set of potential solutions is neither enumerable nor simply representable; (3) The processing to be accomplished is unknown until other processing is done; and (4) An acceptable solution is required within a limited time.

Most design problems, including programming a computer, are of this nature; so are the very complex information processing tasks like translating languages or abstracting scientific articles. The current arguments about thinking machines and general-purpose robots also revolve about whether computers can deal with problems of this general nature.

The problem of playing good chess certainly falls into this class of ultracomplicated problems. It is a useful type case for general discussion because the nature of the task and the complexities surrounding it are common knowledge. Further, it already has something of a history.[1-4]

The aim of this effort, then, is to program a current computer to learn to play good chess. This is the means to understanding more about the kinds of computers, mechanisms, and programs that are necessary to handle ultracomplicated problems. The limitation to current computers provides the constant reminder that the heart of the problem lies in the limitation of resources, both memory and time. This aim would be somewhat ambitious even if all the details were available. The paper will actually be limited to presenting an over-all schema which appears feasible, and relating it to some of the critical problems which must be solved. The reference to learning expresses the conviction that the only way a machine will play good chess is to learn how. Although learning considerations have been prominent in the thinking and motivation behind the machine, attention will have to be restricted to the performance system; that is, to those features which are necessary in order to play the game. However, some of the learning potentialities implicit in the performance system will be discussed.

The work presented here represents the early phases of an attempt to actually construct such a program for the JOHNNIAC, one of Rand's high speed computers.

Before starting it is desirable to give some additional conditions of the problem. From now on the computer with program will be called the "machine," and the various parts of what it does will be called "mechanisms." The problem is not to construct a machine which can induct the rules of chess by playing; it will be instructed concerning the legalities. Finally, the ma chine is only to do the job of one man; it will require an outside opponent, human or otherwise.

PROBLEMS

As everyone knows,[5] it is possible in principle to determine the optimal action for a given chess position. First compute out all continuations to the bitter end. See whether they win, lose, or draw; and then work backwards on the assumption that the opponent will always do what is best for him and you will do what is best for you.

The difficulty, of course, is that this "in principle" solution requires a rather unnerving amount of computing power,[1] and doesn't give any clues about what to do if you don't have it. It will provide us, however, with a short checklist of problems which must be solved if the computing requirements are ever to shrink to a reasonable size.

The most striking feature of the "in principle" solution is the tremendous number of continuations. This is accounted for by both the number of new consequences that appear at each additional move into the future, and the large number of moves required to explore to the end. This provides two problems:

1. The consequences problem, or which of the possibilities that follow from a given proposed action should be examined;

† The Rand Corp., Santa Monica, Calif.

[1] C. E. Shannon, "Programming a computer for playing chess," *Phil. Mag.*, vol. 41, pp. 256–275; March, 1950.

[2] M. Weinberg, "Mechanism in neurosis," *Amer. Scientist*, vol. 39, pp. 74–98; January, 1951.

[3] P. I. Richards, "On game learning machines," *Scientific Mon.*, vol. 74, pp. 201–205; April, 1952.

[4] C. S. Strachey, "Logical or nonmathematical programmes," *Proc. Ass. Computing Machinery*, pp. 46–49; September, 1952.

[5] J. von Neumann and O. Morgenstern, "Theory of Games and Economic Behavior," 2nd Ed., Princeton University Press, Princeton, N. J.; 1947.

2. The horizon problem, or how far ahead to explore.

The possibility that one might stop looking at some intermediate position only raises a third problem:

3. The evaluation problem, or how to recognize a good position when you see one.

Another feature of the "in principle" solution is the identical examination of all the possible alternative actions. Despite the similarity in describing both present and future moves, it is worth while to keep distinct the actions that are actually available at a move, and from which a choice must be made; and the future consequences of these actions, which may include, among other things, limitations on the alternatives available in the future. Hence, we have:

4. The alternatives problem, or which actions are worth considering.

These four problems, consequences, horizon, evaluation, and alternatives, will be sufficient to keep us aware of the difficulties as we search for a set of mechanisms to play chess. Solutions must be found to all of them if the machine is to play good chess with reasonable resources.

Overview

There is a common pattern to the solutions to be described here. In all of them the machine uses very partial and approximate methods. It is as if the machine consisted of a vast collection of rules of thumb. Each rule is a much oversimplified expression of how the machine should behave with respect to some particular aspect of the problem. The rules are of all kinds: chess principles to follow, measurements to make, what to do next, how to interpret rules of thumb, and so on. These are so organized that they form the necessary qualifications and additional specifications for each other. Each rule is essentially a machine program. At any particular instant the machine is under the control of some such rule, or shunting between rules under the control of a master program.

The main effort of the paper is devoted to describing how such a set of rules can be defined and organized to achieve solutions to the four problems, and thus provide a schema for a machine which puts all these pieces together to play chess. Only minor effort is devoted to indicating the detailed structure of these programs at the level of machine code.

One aspect of the underlying coding does require attention, and is dealt with at the end of the paper. The large number of rules, their complexity, and the necessity for adding new ones and modifying old ones, implies the use of a fairly extensive general-purpose language. That is, all these rules are to be given in this language or pseudo-code, as it might also be called. Hence, each use of a rule must be preceded by an interpretive step. However, a few programs suffice for using any and all of the rules that might be required in the machine.

Preliminaries

Let us start by providing the machine with a few basic facilities. Each chessman and each square of the chessboard needs a name, suitably coded into binary bits. A fixed set of addresses are set aside to hold the current position. This can be given as a list of the men with the squares they occupy, including a "zero" square if the man is off the board. The machine can accept an opponent's action by reading a punched card. This can be given as a list of the men which have been moved, along with their new locations. Thus an action involving a capture has two terms: one giving the new location of the man that captured, and a second giving the location of the captured man as the zero square, that is, off the board. The machine obtains the new position by substituting in the old one, which is already stored. It can also output its own action on a punched card.

The machine must also be equipped to answer an array of elementary questions that recur constantly, such as, "Can a given man move to a given square?" or, "What man is blocking the Queen Pawn?" Each of these questions can be answered by a straightforward and not-too-lengthy investigation of the current position. The number of actual programs needed is within reasonable bounds since the more complicated questions are combinations of the more elementary ones.

Assume, then, that the machine has these basic capabilities. They have solved none of the four problems.

Goals and Tactics

Suppose, in the midst of a game, the machine (which is White) has stored in a suitable place the following expression:

$$att(BKB, WKR).$$

The machine interprets this as: "Attack the Black King-Bishop with the White King-Rook." Attack will be considered to mean a successful attack, which in turn means that the attacking man (here, the Rook) is capable of capturing the object of the attack (here, the Bishop) with a relative gain in material after the smoke of engagement has cleared. Thus, for a given position, the attack is either successful or not; and the machine can determine this by a sequence of those elementary programs given it earlier. Call such an expression a goal, which is either achieved or not achieved for any given position.

Now, suppose the machine were to start tracing out continuations. It could determine at each new position it arrived at whether or not the goal was achieved. If it is, the machine could stop searching. This might provide a solution to the horizon problem: have a goal and only explore continuations until a position is reached where the goal is achieved.

The machine can represent such a set of continuations as a branching net, or tree, of actions. Each action is linked by some kind of indexing to the immediately preceding action, and to each action is associated by this indexing a number of other actions that might possibly follow. Such a tree is a tactic for a given goal if it always terminates in positions that achieve that goal. Further, a tactic will always indicate a single action for the machine to take, and many actions for the opponent.

But if the machine has a goal like "attack the Black King," it is right back with the original difficulty of searching for continuations which end in checkmate. This merely indicates that in general one can hardly expect to find a tactic for a given goal. But on the other hand, one sometimes will: men do get captured and games do get won.

By some means let the machine acquire in memory a second goal, which is indexed to the first:

$$blk(BQ, BKB) \rightarrow att(BKB, WKR).$$

The machine interprets this as: "Block the Black Queen's ability to recapture where the Black King-Bishop is." If this ability to recapture was in fact one of the deterrents to taking the Bishop with the Rook, then achievement of this second goal could be considered as an intermediate aid to achievement of the attack. Such a goal will be called a subgoal. It may itself have subgoals, and thus the machine may acquire rather large networks of goals.

Now, instead of trying to construct a tactic for the attack goal, the machine can search for a tactic for the subgoal of blocking. Perhaps it will find one. If it cannot, then it needs either other subgoals for the attack or some subgoals for the block.

The single large search for winning positions has been replaced by two searches, each terminating the other. The machine searches for intermediate goals; it stops this search when it finds subgoals it can achieve. It determines the question of achievement by a search for particular sequences of action. This search terminates with a completed tactic, that is, when the sequences end in goal achievement.

But when to stop if the tactic search continues to be unsuccessful? The mechanism of subgoal formation operates only if the signal has been given to terminate further search for a tactic. Before discussing this, it is appropriate to consider another aspect of the total problem, which, after a twist or two, will lead back to this same decision problem.

Likelihoods

The mechanisms introduced so far have only yielded a solution to the horizon problem. For instance, they do not provide a solution for the consequences problem. It would still seem the machine would examine all branches at any given future move it arrived at. Thus, tactics might be short, but they would have very large spreads.

Suppose the machine had some other goals in its memory labeled "opponent's goals." For example, the machine might have:

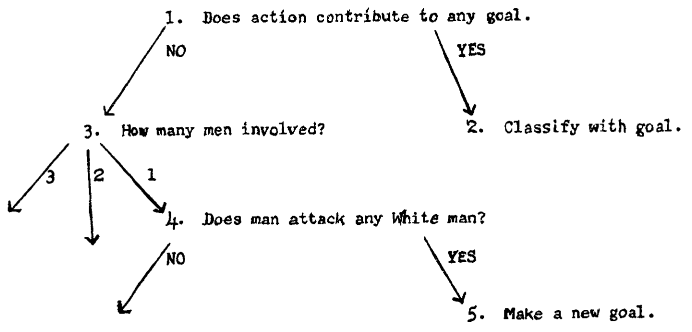
$$ctl(Q5).$$

The machine interprets this as, "Control the Queen five square," and takes as a definition of achievement for the opponent that he could win an engagement if White tried to occupy the square or capture a Black man who did occupy it. If the machine considered this goal to be one of the opponent's important goals, then clearly, any opponent's action which achieved this goal would be relatively likely to occur. Or would it? It certainly depends to what degree the machine's representations of the opponent's goals correspond to the opponent's real goals or whatever other mechanisms guide his actions.

The machine must now be equipped with all the apparatus for collecting evidence, forming hypotheses, verifying them, and making inferences from them. At least in some rudimentary form these will all be required to develop and maintain a running model of the opponent's intentions.

If such a set of mechanisms were available to the machine, then, within certain limits of accuracy, it could assign likelihoods to the various alternatives by inference from the model. These likelihoods provide a solution to half the problem of consequences: how to know which consequences to examine at any position that is the opponent's move. Examine those alternatives which seem most likely.

A rather large number of mechanisms are involved. First, each action that the opponent makes must be classified as stemming from some goal or goals. Somewhere in memory let there be stored some expressions like:



1. Does action contribute to any goal.
NO
YES
3. How many men involved?
2. Classify with goal.
3
2
1
4. Does man attack any White man?
NO
YES
5. Make a new goal.

These expressions are coded in the language spoken of earlier. The machine starts with expression 1 and interprets it. This instructs the machine to test each of the opponent's goals to see if the action contributes to it. This term is used in a narrow sense: only men who can capture contribute to an attack; only men who command the square contribute to its control. Thus, in the example, "control Queen five," the machine would determine if the action had resulted in any new man commanding the Queen five square. If the answer is yes, the machine proceeds to expression 2. When interpreted, this instructs the machine to classify the action as stemming from the goal to which it contributes. This ends the decision process. But as usual, the answer could also be no; the first rule does not suffice to classify the action. The machine then proceeds to expression 3, interprets it, and counts the number of men who changed in the current action. Finding the number to be one, which means that nothing complicated like a capture, promotion, or castle occurred, it proceeds to expression 4. The machine is now instructed to see if the man which moved can be considered as attacking some new man, and if so to read expression 5, which instructs the machine to create a new goal. Notice that this alternative is considered only if the action has failed to be consistent with any of the goals already hypothesized as accounting for the opponent's action.

So far only a mechanism for classification and possible introduction of new goals has been described. It should be remarked about this mechanism that the content of the expressions is much less important at this juncture than the general schema. Other expressions and more of them could just as well be used. Instead of describing the rest of the mechanisms needed to complete this component, it is necessary to move on to other key pieces. The undescribed elements will be constructed in the same vein. For instance, there will be expressions for obtaining likelihoods like, "How often has a goal been contributed to recently," and "High activity implies continued activity is likely."

Utility

Even admitting the machine can assign likelihoods to the opponent's alternatives, this solves only the half of the problem of consequences associated with opponent's moves. There remains the other half: how to know which of the machine's possibilities to explore when searching for a tactic.

Consistent with the design philosophy emerging here, the machine will have a section of memory devoted to expressions in the language that are relevant to this problem. These will be organized into a decision net in the same fashion as the expressions for classifying opponent's actions. The object of such a net will be to assign a utility to future machine actions that could be explored in trying to complete tactics. This utility would be expected to reflect a complex of factors. The feature to note is not just the possibility of such nets, but the kinds of relevant information that now are available to be used.

The most important kind, I think, is that every man has associated with it the functions that it is performing. These associations occur by means of the goal structure. Suppose there is a subgoal that the King-Knight supports the King-Bishop; that is, can retaliate by recapturing any man that captures the Bishop. This serves as a statement that the Knight has a function to perform. Any goal whose tactic depends on moving the Knight into a new position must compete with the goal which already has the Knight supporting the Bishop. The machine can determine this by a simple search of its goals. Therefore a reasonable decision net would recommend exploration of this Knight's moves only as a last resort.

The vision of a large collection of small reasons why various moves should have low utility for exploration raises a caution. We do not want the machine to spend all its time examining the future actions of committed men; yet if it were never to do this, it could overlook real opportunities and might even gradually paralyze itself by the accumulation of commitments. The solution, of course, is the random element. The difficulty with random search in a complex environment is that in reasonable time scales, nothing will ever be discovered. Therefore, randomness should be introduced into the decision nets only in small and well controlled amounts. The machine should rarely search for combinations which sacrifice a Queen and Rook to no apparent gain only to develop a mate eight moves later.

Level of Aspiration

By now we have finally returned to the decision problem stated earlier. There are devices for stopping the search for tactics if successful. There are devices for differentiating the various directions of search. These are likelihoods for the opponent's moves, and utilities for the machine's. But there exists no stop rule in the face of continued difficulties.

To find a solution we turn to a phenomenon well known in the human sciences: levels of aspiration. The fundamental reason why the machine must terminate the search is limitation of resources, in this case mostly computing time. By a level-of-aspiration type solution is meant the introduction into the machine of the limitation of resources as an explicit mechanism. That is, various expressions will exist in the machine which measure and utilize the information about the amount of resources available. Thus, the limitations will no longer function as absolute constraints; the way in which they affect the machine is partially within the machine's control.

For tactics a suitable measure, correlated with computing time but more appropriate to the task, is the likelihood of a position. As the machine explores into the future, selecting one possible action after another, the likelihoods compound like probabilities so that the ultimate likelihood diminishes as actions get further into the future, or are reached through lower-likelihood actions.

The machine sets out on a tactic search with a predetermined level of final likelihood. It gradually extends the tree of actions until each terminal position has fallen below this level; then it stops. If, now, the aggregate likelihood of reaching a position of goal achievement lies above another predetermined aspiration level, the machine considers the tactic adequate. If not, it returns to the goal structure and develops some additional goals, as described earlier.

Transformation Rules

Nothing has yet been said about how all these subgoals are generated. The structure of the mechanism is clear; it will consist of more rules of thumb. There will exist a set of expressions in the language which function like the rules of inference in logic: they allow us to derive new expressions from old ones. They are of the form, "For a goal of type A, try a subgoal of type B." For example, "For att (x, y) try def (y)." This expresses the fact that there is some merit to defending the men who are involved in carrying out an attack. If this transformation rule, as it is called, were applied prior to the opponent's actual attack on the attackers, this would constitute a fine example of anticipatory response.

The machine must examine its goal structure to determine the types of goals for which it requires new subgoals. It selects out those transformation rules which might be relevant; that is, where the structure of the goals in the net fits the specifications of the first part, or premise, of the transformation rule. Several possible rules may be obtained. It seems appropriate to introduce a random element to guide the final choice, since the inference expressed by the rules is rather a loose one. The weights assigned to each rule will be functions of experience, so that rules which work get chosen relatively often.

We have again arrived at a place where the important question is the actual content of these expressions. Certainly, if the rules are poor the machine will play terrible chess. Its operation will bear little relation to the objective game situation. Again, other problems are more important. No solution has yet been described for either the evaluation or the alternatives problem. It will be appropriate to consider evaluations next.

Evaluations

It may seem that somehow the evaluation problem has an implicit solution in the array of mechanisms postulated so far. The problem is to relate a position to the winning of the game. Since the machine uses intermediate goals, the problem now is to relate positions to these subgoals. It seems that the tactics do provide this relation. If a sequence of actions in a tactic results, sometime, in the given position, then the likelihood of achieving that tactic's goal from the position is known. The machine spent time computing it when generating the tactic.

The immediate difficulty with this is that it relates a position to a goal only if it occurs in that goal's tactics. This provides no evaluation for the other goals. Hence, it is only a partial solution since the problem of evaluation is assessing the relation to the ultimate goal, which is winning. In order to make playing possible, the machine has replaced the single remote goal with a large number of more immediate ones. Thus, for the machine the problem is: for any given goal, determine the likelihood of achievement from any given position.[6]

The solution exists for the special case where the position occurs in the tactic of the goal. Consider two other special cases. First, suppose the goal were achieved for the given position. It would receive a likelihood of one if we think of likelihoods as similar to probabilities. Second, suppose a goal were not achieved, had no tactic, and no subgoals. It would receive a likelihood of zero, since its achievement rests solely with coincidence or the opponent, both equally bad bets.

These special cases provide the basis for a solution of the evaluation problem. If the machine generates goals as described, then at any instant all the terminal elements belong to one of the three special cases. This is so since the situations not covered by the special cases all involve a goal having subgoals, which means they cannot be terminal elements. The terminal evaluations are a set of boundary conditions, from which, step by step, evaluations can be assigned throughout the goal structure.

In the general case, the problem will be to compute the likelihood of a goal, given the likelihoods of all its subgoals. It is here that the relation of subgoal is given operational significance. The machine operates as if the following principle were true: if g_1 is a subgoal of g_2, then an increase in the likelihood of achievement of g_1 produces an increase in the likelihood of achievement of g_2. It interprets its experience in this light, so that contrary instances are treated as implying that the subgoal is a poor one.

The laws of combination of likelihoods must have several simple properties, mostly those implied by the principle above. Other than this, it may not make too much difference; simple linear combination may do admirably.

Although some of the details have not been enumerated, the machine now has the mechanisms for evaluation. The result is a large set of numbers, one for each goal. The reason why more combination is not required, or perhaps not even desirable, leads to the problem of alternatives.

Alternatives

Clearly, the machine is interested only in alternatives that further its goals. Now, goals are general statements, and the tactic is the bridge provided to pass between the goals and the very particular current situation. Tactics provide actions that lead to goal achievement. At any moment, then, there are a certain number of tactics and these yield an equal number of alternatives (not necessarily all distinct), all of which are worth considering since they each further some part of the machine's goal structure.

This would be a solution to the alternatives problem if this set were always adequate. It can hardly be expected to be so at all times. For instance, right after a very unexpected move by the opponent there may be no tactics left at all.

The solution lies in more levels of aspiration, partial efforts and iteration. For each alternative, the machine computes the goal evaluations for the new position that would follow if the alternative were chosen. This yields an evaluation of each alternative's effect throughout the goal structure. Although this evaluation appears to be

[6] Throughout the paper I have very carefully used the term "likelihood," instead of "probability." The logical status of the entities referred to is not at all clear since the machine uses the measured likelihoods to determine action which in turn determines whether the move or position will ever occur; that is, what the likelihood "really" is.

only one move deep, it is fundamentally grounded in the tactics, which extend much further into the future.

The decision is now made whether to make the available set suffice, or whether to return and work some more: to add and modify the goals and tactics. This is a level-of-aspiration type of decision, which will depend not only on whether the alternatives are "good enough," but also on how much time remains, and whether the move is crucial. Only if the decision is made not to explore and expand further, is the best alternative picked from the limited set and punched into the card as the machine's actual move.

The term "best alternative," is used in a very casual way. The evaluations consist of many numbers, at least one for each goal. It is clear that if a single alternative dominates all others, it should be chosen. It is also fairly clear that an alternative which achieves a very important subgoal is to be preferred over one which only increases the likelihood of a few very subordinate ones. But basically this is a multiple value situation, and in general no such simple rules can be expected to indicate a single best action. The problem for the machine is not to somehow obtain a magic formula to solve the unsolvable but to make a reasonable choice with least effort and proceed with more productive work. There are other ways to deal with the problem; for instance, include conflict as a fundamental consideration in the decision to explore further.

Thus, at each move the machine can be expected to iterate several times until it achieves an alternative that it likes, or until it runs out of time and thus loses the game by not being smart enough or lucky enough.

Performance Schema

The pieces now exist to give an over-all schema for the performance system of the chess learning machine. This is a set of mechanisms which is sufficient to enable the machine to play chess. There is no learning in this system; it will play no better next time because it played this time. If the content of all the expressions required is appropriate, it will play good chess; if they are not, it will play very poor chess.

This performance system is highly adaptive. A goal structure peculiar to each play of the game is generated during the course of play. Tactics reflect the minute detail of the current situation. This short-run adaptability is not to be confused with learning which would permanently affect the way the machine would play in the future.

Fig. 1 gives the schema of operation. Rather than present as systematic and complete a representation as possible, attention has been given to relating the elements discussed so far. The rectangles represent the major kinds of information in the system. These may be viewed as memories. The arrows indicate processes that operate on one kind of information to produce another. The small writing by these arrows relates these processes to key words used earlier. Some of the main decisions are put in circles, since it makes the diagram easier to follow. The programs for carrying out most of these processes are the various nets, like the classification net. For the sake of clarity, these are not shown as explicit kinds of information, although they certainly occupy a large part of the computer's memory.

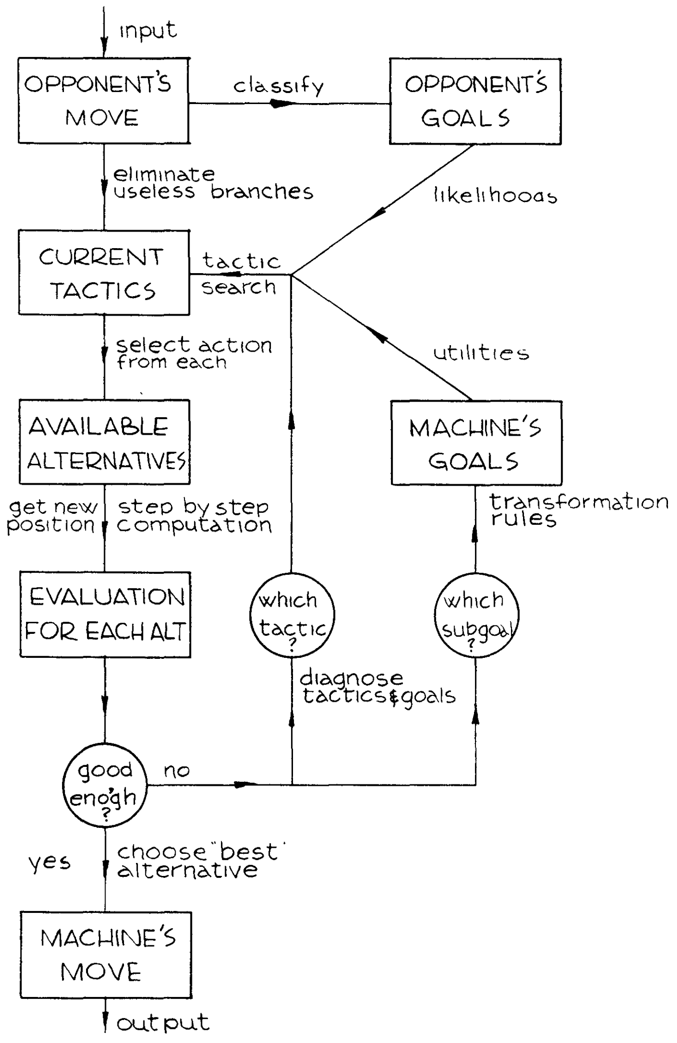


Fig. 1—Schematic flow diagram for performance system.

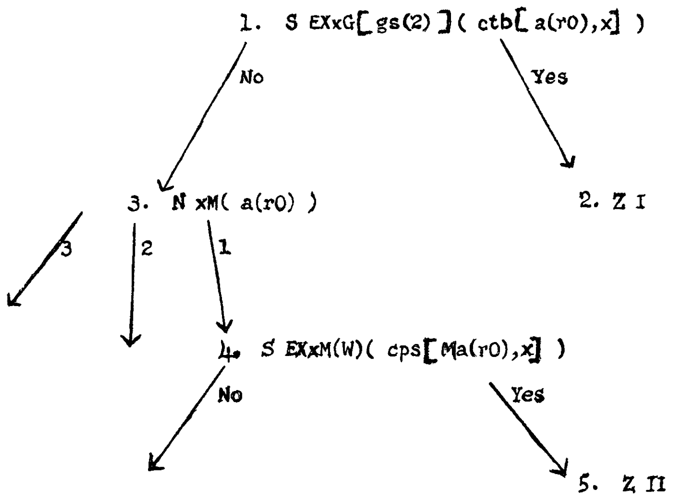
Each sequence always starts with an opponent's move being received (at the top). The process continues (downward) by a series of straightforward computations until the question is reached whether the situation is "good enough." This is the fundamental question. If the answer is yes, the machine has only to choose from among the available alternatives and play, thus ending the sequence (down). So far the effort spent is nominal. If, however, the answer is no, the machine proceeds to the modification and extension of the goals and tactics (to the right and up). This part is of indeterminate duration and effort and utilizes all of the complex apparatus that has been built up. Following this, the machine again attempts to produce a move (downward again). This is the fundamental cycle: to try to decide on a move with little effort, to modify the basis of decision, and to try again. Finally, of course, a move is made and the sequence stops.

Language

It is necessary to add a few comments about the general-purpose language required to make a machine of this nature feasible. An essential feature of this language is its ability to refer to itself as well as to the "external" chess situation. The only languages of this power of expression that have been formalized at all are those used in symbolic logic such as the calculus of propositions.[7] The one illustrated below is an adaptation for computer use.

Each symbol has a binary code, although, for efficiency's sake, not all have different codes. An expression, then, is a long sequence of zeros and ones. The expression is decodable by proceeding from left to right: knowing which symbols have been found so far gives sufficient information to determine how many bits to consider next and what class of symbols it belongs to.

The following illustrates this language. It is the translation of the classification net exhibited earlier.



Consider expression 1 as an example. *In toto* it means, "Does the action contribute to any goal?" A question in the symbolic language is a sentence without its truth value. The machine interprets the missing value as an instruction to determine it; that is, to obtain the answer to the question. Expression 1, then, starts with "S" to identify itself as a sentence. (Expression 3 has an "N," and its value would be a number.) "EX x G" means, "there exists an x which is a goal." The "gs(2)" in parentheses gives additional specifications on the range of x. The machine will only search gs(2) which is the opponent's goal structure. The last parentheses contain the predicate, "contribute" symbolized by "ctb." Its arguments are first, an action, "a(r0)," the "r0" standing for relative time zero so the machine will always look at the current action; and then a goal, "x," which is the variable over which the machine will search.

[7] R. Carnap, "Logical syntax of language," Harcourt, Brace, and Co., New York; 1937.

The machine interpretation and execution of such an expression runs along lines very similar to those used in algebraic coding schemes. The machine has programs for determining the truth or falsity of each individual predicate, here only the "ctb." Truth values are combined, with due regard for brackets, by the laws of "and," "or," and "not," depending on which occur in the expression. The machine interprets "EX" as making the sentence true if for any of the goals considered it obtains "true" for ctb.

Learning

The section is limited to some remarks on the requirements for learning and the potentialities for it that exist in the machine.

The large number of highly interrelated mechanisms involved in the performance of the machine indicates a major difficulty. Consider the hundreds of expressions which each determine a highly specific rule of action. How is it possible to have a set that actually fits together to produce effective chess play? The assumption that man can discern which sets will work seems rather untenable; there are too many effects and hidden consequences. Learning seems to offer a solution.

At any given time the machine has a set of expressions, which work tolerably well over some restricted range of environments. The machine generates a few extensions of or modifications to its current set. These are incorporated if their use provides an improvement in performance. The features to note about this learning process are, first, that it gradually extends the set of expressions, and second, that expressions get admitted only if they "fit in" with the others already there. Thus, given that an appropriate training sequence of games are played, the machine will grow itself a set of expressions of sufficient complexity to play good chess.

Many of the mechanisms necessary to provide this learning are already in the machine. First, it is necessary to measure the effects of one mechanism on another. The machine is already able to perform any test or measurement given by a network of expressions. All that is required is that the expressions refer to the appropriate internal parts of the machine and ask the right questions. The language is powerful enough to do this. These nets also allow diagnosis of which parts of a complex mechanism are causing the undesirable effects. Memory is needed to keep performance records, and mechanisms are needed to collect and classify these records; all of this is already possible. The possibilities for new mechanisms are limited only by the modes of expression of the language, since the specific behavior of the performance system is determined by the content of expressions. Finally, the search for new mechanisms is quite similar to obtaining subgoals for goals. That is, there would be sets of transformation rules to give the types of modifications that might prove useful for a given type of expression.

The problem of sample-size requires mention: How large a sample of experience is necessary to obtain learning? Or better: How much information about the effects of behavior is necessary to successfully modify the behavior? Chess affords a good example of this problem. It is extremely doubtful whether there is enough information in "win, lose, or draw" when referred to the whole play of the game to permit any learning at all over available time scales. There is too much behavior. For learning to take place, each play of the game must yield much more information. This is exactly what is achieved by breaking the problem into components. The unit of success is the goal. If a goal is achieved, its subgoals are reinforced; if not, they are inhibited. (Actually, what is reinforced is the transformation rule that provided the subgoal.) This is so whether the game is ultimately won or lost. Each play gives learning information about each goal that is generated. This also is true of the other kinds of structure: every tactic that is created provides information about the success or failure of tactic search rules; every opponent's action provides information about success or failure of likelihood inferences; and so on. The amount of information relevant to learning increases directly with the number of mechanisms in the chess playing machine.

Conclusion

"As every design engineer knows, the only difference between a good design and the actual machine is time and effort." This adage is a byword in the field of robots and thinking machines. The scheme presented here is not far from a "good design." One can estimate the man-hours necessary to draw up the detailed flow diagrams from which machine coding follows as routine chore. But this is not sufficient. These mechanisms are so complicated that it is impossible to predict whether they will work. The justification for the present article is the intent to see if *in fact* an organized collection of rules of thumb can "pull itself up by its bootstraps" and learn to play good chess.

Comments on Session on Learning Machines

Walter Pitts (Research Lab. for Electronics, M.I.T.) The speakers this morning are all imitators in the sense that the poet in Aristotle "imitates" life. But, whereas Messrs. Farley, Clark, Selfridge, and Dinneen are imitating the nervous system, Mr. Newell prefers to imitate the hierarchy of final causes traditionally called the mind. It will come to the same thing in the end, no doubt, but for the moment I can only leave him to Mr. Miller, and confine my detailed remarks to the others.

It is of great value to the neurophysiologist to have people invent machines that perform some of the same tasks as parts of the nervous system, especially when the machines operate along the same general lines. This would be true at any time, but is especially so now, which I can make evident by a thumbnail sketch of the present state of the science.

Anatomy has divided the brain into internally homogeneous parts, called *nuclei*, has named and classified them, and has traced the more conspicuous pathways, or *fiber-tracts*, connecting them. After adding some substantial contributions from electrophysiology, the result is a *block-diagram of the nervous system.* Although incomplete in detail in most places, particularly with respect to connections employing few, small, or slow fibers, it is still a generally reliable guide to research. Thus, for part of the visual system, anatomy draws a diagram running somewhat like Fig. 1.

The next chief problem is to find out what transformation the arriving information suffers within the nuclei. There are several ways of doing this. The oldest was to remove a particular nucleus in an experimental animal, and watch the animal after recovery to see what capacities it had lost. The limitations of the method are very great. At most, one can infer that a nucleus is necessary for a capacity, never that it is sufficient. The more complicated disabilities often disappear gradually, by a process, such that one cannot tell whether it was the removal of the nucleus or the incidental dis turbance of something else during operation that caused the disability called the "vicarious assumption of function." And the method cannot be applied to men, in whom alone the more complicated forms of thinking can easily be detected. (Accidental human brain damage is rarely well localized, and one never has tested the man befor e the accident.) This line of attack is therefore becoming obsolete, excepto mong experimental psychiatrists wishing to discover the seat of the soul, the unconscious, the emotions, or other metaphysical entities.

A more useful method is to apply various

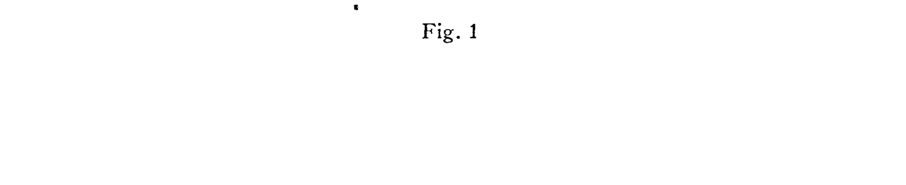


Fig. 1

stimuli to tracts of fibers leading into a nucleus and record from those leading out. This procedure is less generally successful than one might expect. Most fiber tracts are not compact bundles, easily reached for local stimulation and recording; moreover, even if they are, one cannot tell what "information" he has sent in, or what has come out, unless he knows the coding system. A tract will consist of thousands or millions of fibers, each carrying a sequence of impulses in time which is arbitrary except in not having too many impulses too close to one another. In principle, we ought to record the impulses from each fiber separately, since it is possible for the relevant information to lie (1) in the intervals of time between successive impulses in a train proceeding along each single fiber; (2) in which fibers are carrying the impulses; (3) in the intervals in time between impulses, or trains of impulses, in different fibers. To do this is technically impossible, nor could one interpret the results if he had them. In favorable cases, we can do one of two things: record for all the fibers of a tract in parallel (one then cannot tell in which fibers a discharge originates), and record from one fiber in a tract. Both of these give useful results, as we shall see. But such favorable cases are rare. On the whole, they occur in nuclei close to the primary sensory input to the system and the ultimate motor outflow from it. Even for these nuclei, in many cases the signal along a particular output fiber depends upon only a few of the input fibers, and it is quite impossible to find these and isolate them for stimulation.

This brings us to the remaining line of investigation, the direct study of the nuclei themselves. Here we have two kinds of knowledge to start with. The classical histology of Ramon y Cajal and his successor has told us in some detail the structure and mutual arrangements of the neurons in a nucleus and the principal lines of connection among them. The kind of information provided may be seen in Fig. 2, from Ramon y Cajal. One very interesting conclusion may be drawn from drawings like these. The division into layers, the types of cells, their relative abundances, and the general character of their connections are determinate, and largely the same from place to place and brain to brain. A pyramidal cell of the third layer, say, ending on the giant cells of the sixth layer (which give rise to the output fibers) proceeds largely vertically: it ramifies horizontally comparatively little, and ends on the giant cells either directly beneath it or only a little distance away. Within these limitations the connections appear to be random: it is not determined *a priori* to *which* giant cell a pyramid shall project, nor by how many terminal branches, but only that it shall project to some such cells beneath it with a probability declining with lateral distance. This appears to be a general fact about the nervous system: it is never predetermined that a particular cell in a particular place shall project to another particular cell in another particular place, but only that all cells of a given type in a particular locality shall connect (roughly) to cells in another definite locality, with a considerable uncertainty in the boundaries of the "localities." A little reflection on the amount of information required to specify completely the wiring diagram of 10^{10} neurons convinces us that it would be impossible for the chromosomes to specify the connecting of the nervous system much more precisely.

I am speaking here, of course, only of such nuclei as do not learn, or of such others as have not yet learned anything. Learning, of course, may change the character of connections until what was once only statistically determinate becomes minutely so. This fact puts the nervous system in sharp contrast to any ordinary computing machine.

The second kind of knowledge concerns the elementary processes determining the transmission of impulses from neuron to neuron. In a very few places in the nervous system, e.g., the motor cells of the spinal cord, several different bundles of fibers leading directly to the cells can be isolated, and the efferent fibers from the cells constitute a compact and accessible bundle. In such cases, we have been able (see the paper of D. P. C. Lloyd, for example) to find out something about the way the impulses delivered to a particular neuron over several channels combine to determine what it shall do. I ought to emphasize that these neurons are not very typical; most of the neurons in the brain, which differ from these in shape, and geometrical relations to their afferents, may behave differently. In some cases they certainly do: some neurons in the cerebellum and respiratory center fire at a regular rate even when no impulses are reaching them, and respond to stimulation of afferent fibers by changing their frequency for a while. Nevertheless, there is always a reasonable temptation to suppose the neurons we cannot test differ only quantitatively, not qualitatively, from those we can, and even—for some of us—to hope we shall some day be able to guess, by calculating the field of current produced by impulses in afferent fibers arranged in a particular way about the neuron, what its functional relation to these particular afferents might be.

These two kinds of knowledge—partial knowledge about the kinds, positions, and interconnections of the neurons in a nucleus, and partial knowledge of how the activity of a neuron depends on those of its immediate afferents—are what our present investigations of what the nucleus computes are founded on. From this it is perhaps evident what use a high-speed computer can be: if one boldly oversimplifies, and constructs a model network composed of layers of model cells connected statistically as in Fig. 2, say, (after counting the actual density of cells of each layer and the statistics of their connections) and makes reasonable assumptions about the elementary synaptic relations, he can use the computer to calculate the behavior of the entire "model nucleus," in exactly the same way Messrs. Farley and Clark have done (without attempting to imitate the known anatomy of any actual nucleus).

Now the construction of such models and computation of their behavior will not be very useful in understanding the real brain unless the results can be compared, rather directly, with experimental measurements. Fortunately, the last few years have seen the development of two methods capable of providing a comparison with observation.

The first is the development of the ultramicroelectrodes capable of penetrating into a single neuron, or even a single fiber, in such a way as to record its discharge independent of all its neighbors. One cannot thereafter make an anatomical picture like Fig. 2 showing *which* cell one was recording from, and *its* particular connections, but one can tell which layer it was in. It is then possible, with a large enough sample, to find out how the discharge of cells in that layer depends upon a stimulus applied to the sense organs or some other tract or nucleus afferent to the one studied. This technique provides one direct way of checking the computed conduct of someone's model for that nucleus.

It is unfortunate that present technique allows us to insert only one such electrode at a time, so that we cannot follow activity from one layer of the cortex to another, or analyze adequately such structures as the primary visual cortex, where different parts of the visual image on the retina are projected simultaneously on different parts of that cortical area in a sort of distorted map.

The second method is a semi-microscopic one. It endeavors to follow the transmission of a group of impulses from place to place within a nucleus, or along a group of fibers connecting one part to another, without recording from individual units. This procedure also requires microelectrodes, but is not so fine as the other. The object is to secure a record of changes in the general potential field at each of a large number of points, without having the records unduly perturbed by a cell or fiber immediately adjacent to the electrode tip. This potential field cannot be interpreted directly in terms of nervous activity (although many of the older electro-physiologists thought it could), because the neurons are immersed in a good conductor, and it usually happens that activity of any part of a nucleus generates currents and potentials observable throughout the entire nucleus. When several parts, or all of them, as is usually the case, are simultaneously active, the potential at all points depends upon the activity at all points, and the distribution of activity cannot be read off from the potentials by inspection. In these cases one must compute the divergence of the potential field, which does represent the distribution of activity. (An impulse in a neuron consists of a region on its bounding surface where current is being absorbed from the external medium into the cell, flanked by regions where it emits current to the outside: for the potential field measured in the external medium, therefore, an impulse acts like a sink of current flanked by sources. The divergence of the field is therefore a measure of the number of impulses at a given place and time.)

It is obvious that this method likewise can provide a direct experimental check of computations made with models. It too has considerable limitations in that the simultaneous potential field required cannot be obtained: one must record from different points of the nucleus seriatim, repeating the stimuli along the afferent channels as exactly as possible, and taking all possible precautions to detect any change in the condition of the nucleus during measurement.

These new experimental methods put a new complexion on the problem of understanding the nervous system. Formerly, the invention of models and computation of their properties was not of very much use. Almost any form of gross behavior, from maintaining a standing posture to perceiving shapes or following movements with the eyes, involves at least six or seven interconnected nuclei. Then one had to invent six or seven models at once, making the doubtful assumptions about synaptic function and detailed histology necessary to supplement our actual knowledge of each nucleus, and then making more assumptions to complete what was known about their interconnections. When one looked for no more precise and detailed an experimental check than an ultimate comparison of the calculated behavior with the actual behavior, the piling-up of dubious hypotheses on one another was so great that, even if the final result should then agree, the hypotheses would gain so little verisimilitude that rational men hardly cared to venture so forlorn an enterprise at all.

But now, as we begin to trace impulses through a nucleus, we can divide the problem, and turn our energies to making separately a model of each nucleus which agrees with single-cell recordings and field-mapping on that nucleus. In this enterprise, with the help of the high-speed computer, there seems a reasonable hope of success.

We ought, therefore, to persuade Messrs. Farley and Clark to abandon their Laplacian indifference, their equally probable connections of everything to everything else, in favor of determining the connections among their model neurons, still randomly, but according to more interesting probability distributions. I have reasons for thinking that this possible generosity of theirs might not be ill-advised for their own secret purpose. (I suppose everyone else has guessed it as I have: they mean to invent a cheaper and more efficient substitute for Generals of the Air Force; and one must not forget that these latter are organisms with nervous systems.)

More seriously, though, it will be exciting to be the neurophysiologist if they repeat very many times the experiment of letting their random net acquire a definite structure by learning and can then discover what is common in the "effectively structural" changes in the network so produced; we might imagine something new to look for in the erudite brain to distinguish it from the unlearned.

Messrs. Selfridge and Dinneen are less humble imitators of the animal nervous system, particularly in their learning mechanism; but in their elementary operations they follow it very closely indeed. There are two kinds of experimental results which seem interesting to mention here. The first concerns the response of single fibers in the optic nerve, as recorded by microelectrodes, to a flash of light suddenly appearing to the whole retina. It turns out that the fibers fall into three principal groups: the "on-off" fibers, which respond by a short train of impulses when the light is turned on, and another when it is turned off; the "on-and-during," the discharge of which continues throughout the presence of the light; and the "off" fibers, which respond only after the light is turned off, and then continue for some while. In cats, and probably in primates, the vast majority of fibers are of the first, or "on-off" variety, and therefore respond principally to changes in the degree of illumination, rather than to illumination itself. This suggested an explanation of the well-known fact that we see boundaries between light and dark regions far more prominently than what is inside them; indeed, we can even recognize a line-drawing of a face or landscape, a feat nobody would ever believe possible on the basis of simple considerations of information-theory. The suggestion arises from the so-called "physiological nystagmus," or continuous fine tremor of the eyes that goes on all the time. This causes the boundaries of regions of different brightness to pass back and forth continually over the receptors situated near it, so that they are exposed to continuous changes of illumination, and will respond strongly. The receptors situated in the middle of regions of nearly uniform brightness will experience little change, and therefore soon cease responding, except for the relatively few "on-and-during" fibers.

This provides in itself a good correlate of Selfridge and Dinneen's "edging" operation, and their "averaging" is represented by the simple fact that one fiber in the optic nerve responds to illumination of a fair-sized "patch" of receptors. Anatomical knowledge of the successive convergence of fibers in higher nuclei provides ample opportunity, under the simplest assumptions, to provide an iteration of these operations, but to what extent this actually occurs we cannot yet say.

Here we might leave the matter, but the recent experiments of Kuffler provide so much more direct a correlate to these operations, in the retina itself, that I must say a little about them. Kuffler succeeded in recording from a single ganglion cell in the retina—these cells are the origin of the fibers of the optic nerve; the receptors (rods and cones) are connected to them indirectly—and explored, with a very small spot of light, the effect of illuminating receptors at various distances from it on its response. He found that all the receptors in a certain "receptor-field" surrounding the ganglion cell could excite it; but the ones in the center of the field close to the cell affected it quite oppositely from the ones farther away. The "on-center" ganglion cells, as he called them, gave an "on"-response when he illuminated only the center of its field, an "off"-response when he illuminated only the periphery. The "off-center" ganglion cells behaved in exactly the reverse fashion. Illuminating at intermediate distances generally caused an "on-off" response. He also discovered that the receptors in the center of the receptor-field and those at the periphery generally inhibited one another; the effect on the ganglion cell, that is, if both were illuminated, was that its characteristic response to each group was reduced or abolished.

When we recall that these "receptor-fields" for different ganglion cells form a set of overlapping patches covering the whole retina, and that the ganglion cells are of two opposite types, the general result becomes rather complicated, particularly when one includes the "physiological nystagmus." Nevertheless, it is apparent that there is a premium on inequality of illumination, that is, that such ganglion cells as respond most have the greatest inequality of illumination between the receptors immediately adjacent to them and those at a little distance, so that the net result must certainly be a large degree of "edging" as well as "averaging."

It is worth reminding you, in conclusion, that these resemblances between work on the neurophysiology of vision and Messrs. Selfridge and Dinneen's fundamental operations on the alphabet are not at all wonderful or surprising. They know these physiological facts as well as I do, and have deliberately designed their proposed machine to imitate them. Nor is this at all unreasonable, when one reflects that they mean to imitate a human activity, so that it is merely economical to begin by throwing away, as soon as possible, that vast bulk of "information" present in every visual image, which *we* begin by throwing away.

I had meant here to reflect on certain fundamental presuppositions determining the thinking of man, and apparently all other mammals as well, which a machine capable of inductive reasoning offers the first hope of really escaping, and therefore understanding. These are the epistemological limitations first discovered by the philosopher Kant. These "synthetic *a priori* categories of the apperceptive dialectic" constitute perhaps the most fundamental problem of neurophysiology and psychology. Consideration of them impelled Helmholtz to initiate these sciences, and has inspired his successors continuously, until the present. At the moment I can only explain this rather cryptic remark by a reference to the contribution by Heinrich Kluver to the Cold Spring Harbor Symposium, Vol. VII, Visual Mechanisms.

G. A. Miller (Assoc. Prof. Psychology, Harvard University). I want to take this chance to congratulate the speakers in this session for the excellent quality and clarity of their presentations of some rather complex research.

I once heard Robert Oppenheimer, who has a rare gift for the appropriate expression, say that good science is just that work that your colleagues are grateful to you for doing. On that basis, at least, I am sure that I have been listening to good science here this morning—it must be good, because I feel deeply grateful to these men for doing it.

What I want to do here today is to look at what these men are doing from the point of view of a psychologist. Now as a psychologist, concerned with understanding the behavior and the minds of living organisms, I have a lot of problems. In fact, I sometimes think I have nothing *but* problems. So when I see other people trying to solve some of these difficult problems for me, I am inclined to become almost pathetically grateful.

Broadly speaking, psychologists have two somewhat different classes of problems which, for simplicity, I will call descriptive problems and functional problems. A descriptive problem is one in which we try to state the properties that characterize an animal's behavior. Thus we speak of a man as being *honest*, as having *self-esteem*, as being *intelligent*, *well-adjusted*, *introverted*, *dominant*, etc. These same properties can be, and often are, attributed to machines. We say that our automobile is stubborn for the same reasons we say a mule or a man is stubborn—it won't do what we want it to. Such properties are often attributed to computing machines.

I have heard a certain amount of discussion as to whether or not it is proper to ascribe such human properties to computing machines. To my mind, these arguments are completely irrelevant. No matter which side wins the argument, the practical business of designing new machines is not changed in any way. So I will not waste my time arguing over distinctions that have no consequences.

When we come to the functional problems, however, the situation is far more interesting. A functional problem is one in which we try to state how an organism performs a certain function. Thus we speak of a man *detecting* a faint sound, *recognizing* an object, *learning* a skill, *remembering* a name, *solving* a problem, *performing* a job, or *communicating* an idea. These are all psychological functions and we would like to know how it is that living organisms carry them out.

With regard to such functional problems there is a long history in psychology of interest in machines, in robots, that can perform the same functions. Descartes may have been the first to think in this way. The value of such analogies for the psychologist is that they often help to clarify his thinking. If he can express the problem in terms of a machine, then he feels that he has defined his terms scientifically and has eliminated any possible mysticism or magic from his discussion.

A discussion phrased in the ordinary language of everyday conversation is often very ambiguous. In order to be more precise we often have recourse to the language of mathematics where we must define our terms clearly and perform our deductions according to established patterns of logic. Psychologists, however, are often unable to make a precise mathematical statement, and so they have often resorted to this dodge, this halfway house on the road to precision, the conceptual model. A model is certainly superior to a purely verbal argument, and is inferior to a complete mathematical theory. Thus many psychologists have tried to design machines that would perform psychological functions.

This morning we have heard three attempts to design machines for carrying out psychological functions. Selfridge and Dinneen have a machine that recognizes objects. Farley and Clark have a machine that learns by being rewarded. Newell has a machine that solves problems in chess. In all these cases, the psychologist expects to learn whether or not such machines, designed according to certain stated principles, actually perform the functions intended. In that way we get an indirect verification of the adequacy of our psychological postulates.

For a concrete example, consider the question of recognizing objects. In trying to conceive how a machine might do this job, many psychologists have imagined that some memory trace of every object ever perceived in the past is stored somewhere in the nervous system. Then when a new object is presented, its image is correlated with all the stored images, and that stored image which gives the highest correlation with the perception of the present object determines what we call it. However, when we in fact get around to really building such a machine, the inadequacy of such a model quickly becomes apparent. The machine cannot remember so much and it cannot compute that many correlations in the time allowed. Therefore, we are at once forced to perform certain operations on the image that extract from it the significant features, and these features are then compared with a much smaller memory. Such a machine is practical to build, it is in better conformity with the known facts of human perception, and Selfridge and Dinneen have shown that it can work to give the same results.

Similar remarks are also true of the chess machine. It is not good machine design to explore all possible consequences of all possible responses to a problem. And it is not good psychological theory either. So Newell considers a machine that forms inductions and remembers rules and does not try to store away every item of information it has received.

To me, a psychologist, this is the principal lesson to be learned from the papers presented here today. Our problem, our joint problem, is to discover what transformations must be made on the available data in order to preserve intact the significant features and to discard the irrelevant details. When we have discovered such transformations or operations, we shall be a great deal closer to building better machines, and also a great deal closer to understanding human behavior.

A New Nondestructive Read for Magnetic Cores

R. THORENSEN† AND W. R. ARSENAULT†

Summary—**A simple but effective method has been developed for reading information from ferrite storage cores without destroying the information. A single ferrite core containing a small radial hole is used per stored bit. A wire is threaded through the hole and a current pulse is passed down the wire to interrogate the core. The interrogation current temporarily disturbs the residual flux in the core and induces a signal in an output winding. The output signal is unique for the direction of the residual flux and is independent of the direction of the interrogation current.**

The read operation can be extremely fast. Complete read signals may be as short as 0.3 microsecond, making interrogation cycles of 0.5 microsecond feasible. Laboratory tests have proven these cores to have all the properties associated with normal coincident current memories plus nondestructive readability.

Fabrication of these cores should prove practical and economical as compared to nondestructive read arrangements known to the authors. Normal square loop ferrite cores can be altered to the new design or it should be possible to fabricate the radial hole at the time of manufacture.

† Res. Labs., The Magnavox Co., Los Angeles, Calif.

Introduction

THE APPLICATION of ferrites to the electronic computer field for storing information is well known and has been well covered in the literature. For those not completely familiar with present methods of inserting and removing information from such storage systems, a brief review is included.

Fig. 1 shows a toroidal core and its hysteresis properties. Fig. 1(b) is a somewhat idealized curve of a square loop ferrite. The core is shown with three windings, two carrying switching current and the third, a sensing signal. A current of $(-I)$ will bring the core to a point marked $(-\phi_m)$ on the hysteresis loop and upon removal of $(-I)$ the core will come to rest at the point 1. The core is then said to store a binary 1. In order to sense the information stored in the core, a current of (I) is applied. The flux in the core then traverses the loop to (0). The

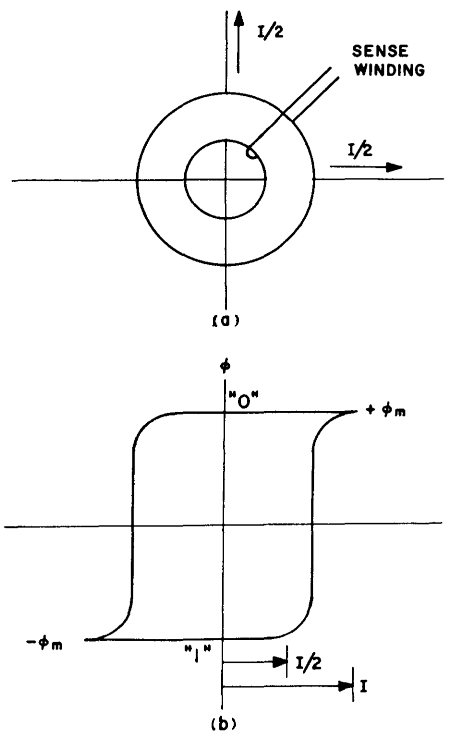


Fig. 1—Coincident current core and hysteresis properties.

flux change is sensed and interpreted as a read signal of 1. If the core had been in the state indicated by 0, there would have been only a very small flux change and the read signal would have been interpreted as a 0. It should be noted from the curve that a current of either plus or minus $I/2$ will not influence the state of the core. This property is used to set up a selection plan in a large memory array. A particular core is selected by the coincidence of two half currents.

A characteristic of the memory systems described is that the reading or sensing process is destructive. If the information is to be retained, it must be temporarily stored by external means and rewritten into the cores. This involves two complete flux turnover times of the core plus the time involved to set up the associated switching circuits.

The access time of present coincident current core memories varies from approximately 6 to 12 microseconds. Some special applications of computers require references to the memory at a much greater rate than is available with the coincident current type memory. Since the turnover time of the ferrite cores was already being pushed to their upper useful limit, engineers started to look for nondestructive methods of sensing the information stored in the core. If this nondestructive reading could be accomplished, the time to rewrite the information could be eliminated.

Two published methods[1,2] accomplishing nondestructive reading are shown in Fig. 2. One uses an auxiliary core or pole pieces to produce a magnetic field in quadrature to the residual flux in the storage core. So long as this secondary flux is below some limit, it interacts with the remanent flux only momentarily and the state or direction of the remanent flux can be detected. Fig. 2(b) shows a method of generating a quadrature field internally in a metallic core. The interrogation current is sent down the metallic ribbon, producing a flux perpendicular to the residual flux since this flux, too, is in the direction of the ribbon.

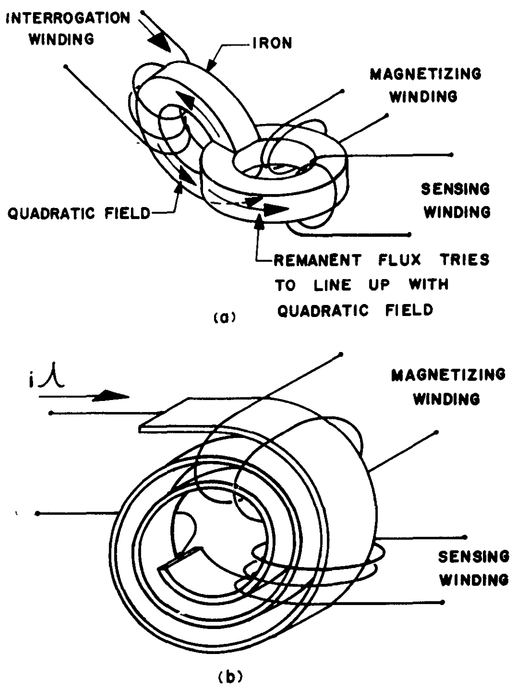


Fig. 2—Quadrature field nondestructive read.

Description of New Read Method

The authors' method of nondestructive read-out is based on generating internally in the ferrite storage core a secondary field which interacts with the main residual storage flux of the core. The interaction is such as to cause a transient change in the residual flux, the direction or polarity of the change depending solely on the direction of the residual magnetization around the core. The flux change is transient as opposed to permanent; upon the removal of the interacting magnetic field the orientation of the residual flux around the core has not been changed nor will it change regardless of the number of times the interrogation process is repeated. Further, since the sign of the transient flux change depends on the orientation of the residual flux in the core, the process may be used to read out information in a nondestructive manner.

The manner in which the secondary field is introduced is illustrated in Fig. 3 which shows a toroidal storage core with two holes drilled on a diameter. A wire is threaded through these holes and a pulse of current is passed down the line to interrogate the core. The flux generated by this current temporarily disturbs the

[1] D. A. Buck and W. T. Frank, "Non-destructive Sensing of Magnetic Cores," AIEE Tech. Paper 53-409; October, 1953.
[2] A. Papoulis, "The Nondestructive read-out of magnetic cores," Proc. IRE, vol. 42, pp. 1283-1288; August, 1954.

residual flux ϕ_R in the core. The change of flux during this period induces a voltage in the sensing winding and the state of the core can be detected.

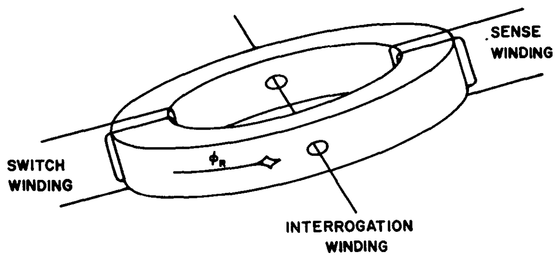


Fig. 3—Core configuration for nondestructive read.

The voltage induced in the sensing winding is shown in Fig. 4. The top line of Fig. 4 shows the interrogation current pulse. This pulse is approximately 0.2 microsecond wide at the base. The second line shows an output signal when the residual flux in the core is in a direction indicated by (ϕ_R) (Fig. 3). The read signal—including recovery—takes only 0.4 microsecond. The switch winding shown in Fig. 3 may be used to change the direction of the residual flux in the core. When the residual flux (ϕ_R) is oriented in the opposite direction from that indicated in Fig. 3, the output at the sensing winding upon the application of an interrogation pulse is that shown in the third line of Fig. 4. The signals shown on lines two and three can be interpreted unambiguously as the storage of a 1 and a 0, respectively.

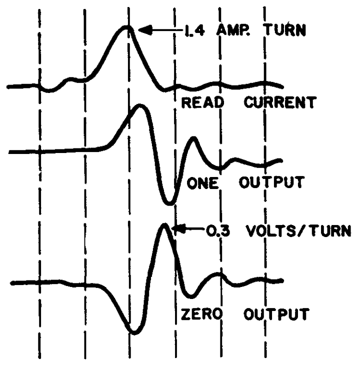


Fig. 4—ZERO and ONE output. Sweep: 0.1 microsecond/division.

Fig. 5 shows an enlarged section of the storage core with the radial hole. The interrogation winding *d, e* is arranged in such a manner that a current through it will produce a flux of one direction in the upper section and of the opposite direction in the lower section. If we assume, for the moment, that the residual flux (ϕ_R) temporarily decreases in magnitude in the presence of the field set up by the interrogation current, it is easy to see that the read signal would be unique for the particular direction of (ϕ_R). The output E is proportional to $(-d\phi)/dt$. If (ϕ_R) is positive and decreases and increases respectively upon application and removal of an interrogation current, the output voltage E will initially go positive and then negative. In the complementary case, (ϕ_R) is negative and an output of opposite phase may be expected. Moreover, the output will be independent of the direction of the interrogation current since the arrangement is purely symmetrical.

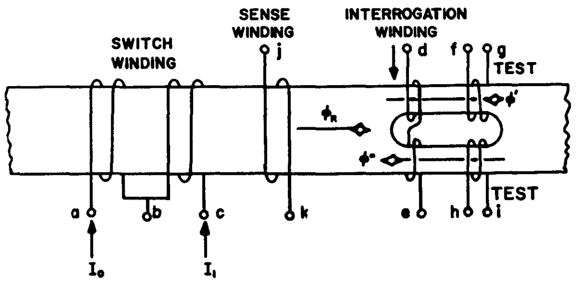


Fig. 5—Windings arrangement for test core.

Explanation of the Read Phenomena

The method of influencing the residual flux in the core by the application of a current through the radially oriented hole may be explained as the application of an influencing flux parallel to, but always opposing, the residual flux. Referring to Fig. 5: the core is first saturated in the direction indicated by (ϕ_R) by passing a current I_1 through the switch winding. If an interrogation pulse of the polarity indicated is now applied, it constitutes a magnetomotive force which tends to force a flux (ϕ') through the upper leg and a flux (ϕ'') through the lower leg of the core. The core however is already saturated in the direction of (ϕ_R) so that the mmf generated by the ampere-turns in the upper leg acts through a high reluctance path. The mmf acting on the lower leg opposes the residual flux and is therefore capable of causing a considerable flux change in the core. In the case illustrated it is (ϕ'') that temporarily reduces (ϕ_R) which in turn causes an output signal to be induced in the sensing winding. If ϕ_R were initially oriented in the opposite direction, (ϕ') would oppose it upon application of an interrogation pulse and cause a signal of opposite phase to be induced in the output winding.

Upon the application and subsequent removal of an interrogation pulse, the residual flux does not return to its initial value. This action is depicted in the hysteresis curve of Fig. 6. Assume the core was initially at point *a* and an interrogation pulse is applied. The flux path might traverse a path indicated by *a*, *b*, and *c*. Note that point *c* is somewhat less than *a* in flux value. Upon the application of a second interrogation pulse, the path traversed would be that indicated by *c*, *d*, and *e*. Again *e* is somewhat less in value than *c*. Upon repeated application of interrogation pulses, the flux in the core will stabilize and traverse some closed path indicated by *f*, *g*. The exact location and size of this minor loop is dependent upon the interrogation pulse amplitude and width, and upon the properties of the core. Subsequent applications of interrogation pulses will produce outputs indefinitely with no further decrease in flux. These signals are represented in Fig. 4. Laboratory tests have shown that the signals settle down in from 2 to approximately 20 pulses, depending on the properties mentioned above; output amplitudes can be made to vary less than 10 per cent during the initial deterioration.

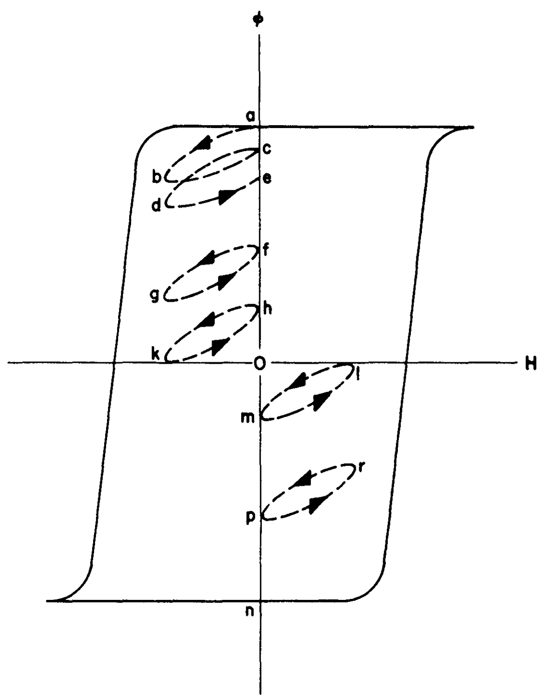


Fig. 6—Hysteresis properties of core showing flux traversions during read.

In the complementary case (where the core is initially in the state represented by point *n*, Fig. 6), the loop traversed during a read operation will stabilize at a path represented by *p*, *r*. It is seen that $\Delta\phi/\Delta t$ is opposite from that which takes place at *f*, *g*.

Laboratory experiments showed further that the information held in the core could not be destroyed, regardless of wide variations in amplitude and duration of the interrogation pulse current. Referring again to Fig. 5, it can be seen that the extreme case of high interrogation current forces the upper leg of the core associated with the radial hole to become saturated in the direction indicated by ϕ' while the lower leg is saturated in the direction indicated by ϕ''. Under this condition, assuming that there is no leakage flux and that the radial hole is oriented in the center of the core, the net flux around the toroid will be zero. Assuming further that the core was initially at point *a* (Fig. 6) this zero flux condition is represented by point *k*. Upon the removal of the interrogation current, the flux will return to the point *h*. Thus, regardless of the amplitude of the interrogation current, the flux in the core can go only to zero and never reverse. Because of this, the elastic properties of the material can always return the core to some point *h* which is in the same direction as *a*. Succeeding interrogation pulses will cause the core to traverse a loop *h*, *k* which is very similar to *f*, *g*. Thus, no information has been lost and readable signals may be derived indefinitely.

Again, in the complementary case, the core starts at *n* and large interrogation currents drive it to a loop represented by *m*, *l*. The distance from the origin to *k* or *l* represents the net field that the interrogation current must apply to the toroid to bring the flux to zero. The direction, whether it be *o*, *k*, or *o*, *l*, is determined solely by the direction of the initial residual flux in the core.

Experimental Results

For test purposes several General Ceramics size F-262 toroidal cores were drilled out as shown in Fig. 3. The toroid is approximately 0.37 inch O.D., 0.18 inch I.D., and 0.12 inch thick and of S-3 material. The radial holes drilled were approximately 0.040 inch diameter.

Fig. 5 shows the winding configuration put on the core for test purposes. *a*, *b*, and *c* forms the switch winding for setting the state of the core. A current I_0 is passed from *a* to *b* in order to set the core in the 0 state while application of I_1 from *c* to *b* will put the core in the 1 state. The third winding around the core (*j*, *k*) is used as a sensing winding. One winding (*d*, *e*) was put through the radial hole as the interrogation or read current winding. Two others (*f*, *g* and *h*, *i*) were put on as test windings. The tests carried out on the cores were of a twofold purpose. First, the presence of the nondestructive read phenomena had to be proved. Then other properties of the cores required for memory application had to be tested while being influenced by the read method. The second purpose was to determine the nature of the read method and ascertain the amount and limit of the residual flux reduction in the core as a result of the read method.

Nondestructive Read Tests

A test cycle was applied to the core as follows:

1. Switch to 0 state.
2. Switch to 1 state.
3. Apply 20 interrogation pulses.
4. Switch to 1 state.
5. Switch to 0 state.

The switch signals applied were 1.8 ampere-turns of 4 microseconds duration while the interrogation pulses were 1.5 ampere-turns and 0.2 microsecond duration.

During step 3 stated above, the output signals may be inspected and the amount of deterioration as a function of the number of interrogations may be determined. Fig. 7 shows typical read signals for the core in the 1 state. The first signal after switching to the 1 state (curve 1) and the twentieth signal (curve 2) are superimposed. The first signal is slightly larger with little undershoot. Since ϕ is proportional to $\int E dt$, the area under this curve represents the flux change in the core. Since the area above the reference line is greater than that below, there must be a net flux decrease in the core during the first read pulse. This is represented by the path *a*, *b*, *c* of Fig. 6. The twentieth pulse (curve 2) has a greater undershoot and the area above the line approximates the area below. This is represented by the path *f*, *g*, *f*, of Fig. 6, and represents no net flux change in the core. It should be noted (Fig. 7) that the amplitude decreases very little with reduction of residual flux in the core. The undershoot or recovery merely increases as the core reaches a stable condition.

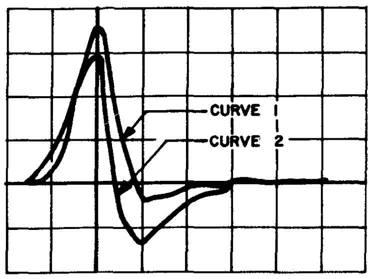


Fig. 7—First and twentieth read signals of a series. Sweep: 0.1 microsecond/division.

The amount of reduction of flux in the core as a result of the interrogation pulses may be measured during step 4 of the program outlined above. During step 2, the core was set to the 1 state from a 0 state. During step 4 the core is set back to the 1 state after having interrogation pulses applied. Fig. 8 shows the signals derived at the sense winding *j*, *k* (Fig. 5) during these periods. The largest signal (curve 1) is the result of step 2, i.e., switching from the 0 state to the 1 state and the area under this curve represents the flux change in the core when going from one orientation to the other. The intermediate curve (curve 2) is the signal derived during step 4, and the area under it represents the flux necessary to restore the core to its original condition after interrogation. If the area under the intermediate curve were just one-half that under the larger curve, the core would have been in a neutral state after interrogation. Graphical integration shows flux area to be approximately one-third of that under the larger curve so that the residual flux actually has been reduced by approximately two-thirds due to interrogation pulses under these conditions.

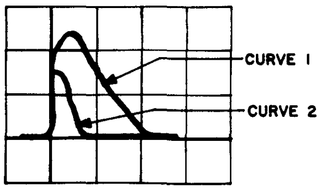


Fig. 8—Turnover signal (curve 1) and reset signal (cruve 2) of core.

Fig. 9 shows the signals derived at the test windings *f*, *g*, and *h*, *i* (Fig. 5), during the time that curve 2 of Fig. 8 was taken. One winding shows a considerable flux change while the other shows none. Therefore all the reduction of flux that took place during the interrogation process occurred in one leg of the core only. In particular, the reduction occurred in the leg in which the mmf set up by the interrogation current opposed the residual flux in the core. The area under curve 1 of Fig. 9 is the same as that under curve 2 of Fig. 8, since both have been shown to represent the same flux change.

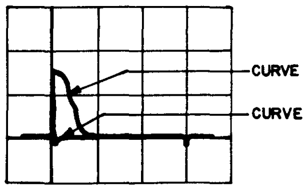


Fig. 9—Reset signals of each leg of core.

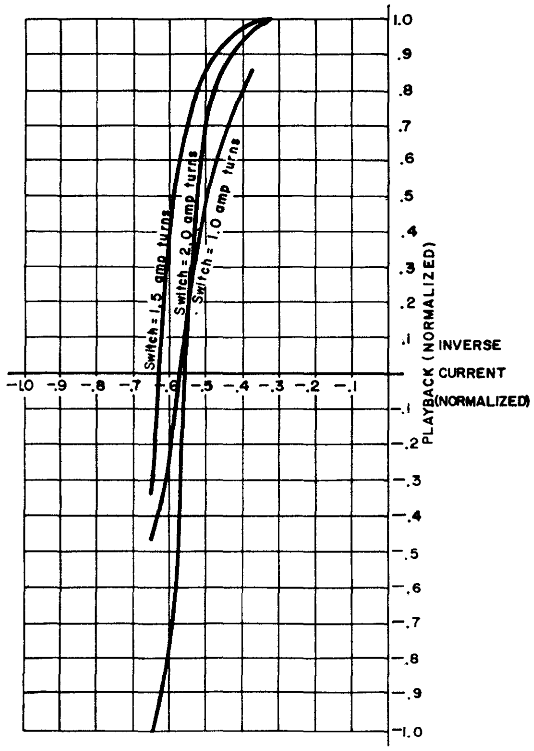


Fig. 10—Switching properties of cores.

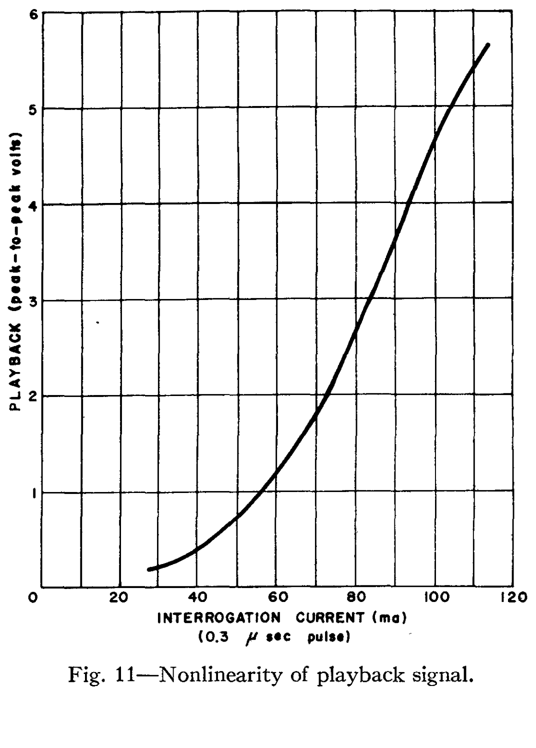


Fig. 11—Nonlinearity of playback signal.

Switching Properties of the Core

Drilling a radial hole in the core and partially destroying the residual flux in the core due to interrogation currents left some question as to whether the square loop properties were still in evidence. These properties are necessary for writing information into the core using a coincident current scheme. Fig. 10 shows three normalized curves for read output vs reset currents. These curves indicate what could be expected in the way of output signals after half currents have been applied to the core in the opposite direction.

It is evident that there is an optimum switch current that gives the best switching ratio. These curves were taken for only a single reverse current pulse. Separate tests showed that with a semi-infinite number of reverse pulses of 6/10 of the setting current, the signal output decreased approximately 30 per cent in amplitude and then no further. This latter test was performed under the optimum conditions indicated by Fig. 10.

Read Output vs Read Current

In the event that some form of core switch were to supply the interrogation current (Fig. 5), there would be no guarantee that all currents on all interrogation lines would be zero except the selected one. Currents from half-disturbed cores could be significant and any noise produced in the nonselected storage cores would be accumulative. Therefore it is desirable to have some nonlinearity in the signal response to the read current.

Fig. 11 shows a plot of signal voltage vs read current for an optimum switch current condition. If we were to choose 90 ma as the read current, a variation of 3 to 1 in read current would give a variation of 18 to 1 in output signal. This desirable effect indicated that selection of read windings might well be accomplished with core switching.

The Electrographic Recording Technique

HERMAN EPSTEIN†

Summary—**A set of criteria for evaluating recording techniques is reviewed. Features, applications, and the basic technique of electrographic recording are discussed.**

Introduction

ELECTROGRAPHIC recording is one of the results in the continuing search for advanced recording techniques. This presentation will be concerned with the technique, as such, and possible applications to computer technology. Additional applications will be mentioned briefly.

One of the early phases in the investigation of recording techniques in the Burroughs Laboratories was a study to establish bench-marks and criteria for evaluating new developments in this field. As a result of this study the following representative criteria were chosen to compare relative merits of the developments in the laboratory as far as automatic data handling machine output is concerned:

1. Type of printer: (a) Serial input—serial output; (b) Serial input—parallel output; (c) Parallel input—serial output; (d) Parallel input—parallel output.—From the point of view of a minimum amount of buffering and control circuitry, (a) is optimum.
2. Choice of characters: (a) Numerical; (b) Alphameric.—A full choice up to 64 characters should be possible, rather than a limitation to the digits alone.
3. Speed of printing: (a) Characters per second; (b) Lines per second.—A technique which offers high- and low-speed recording capabilities in low-cost embodiments is desired.
4. Lines per inch.
5. Character size.
6. Characters per lineal inch.—The technique should not significantly limit the requirements listed under 4, 5, and 6.
7. Line feed: (a) Intermittent; (b) Continuous.—Continuously moving paper during the printing cycles is normally a definite advantage.
8. Copies.—The ability to prepare copies is necessary in many applications.
9. Recording medium: (a) Type; (b) Durability; (c) Acceptance; (d) Cost; (e) Unique characteristics.—Normally, these criteria indicate that preferred media be similar to an ordinary paper stock, rather than specially prepared materials.
10. Character formation: (a) Whole; (b) Matrix.—Depending upon the point of view, one or the other of these character formations is preferred.
11. Actuator of actual recording mechanisms: (a) Electromechanical; (b) Electric pulse; (c) Light flash; (d) Motor driven; (e) Other.—There are more inherent limitations in (a), (c), and (d) than in (b).

† Burroughs Corp., Paoli, Pa.

12. Selection of the printing actuator: (a) Electromechanical; (b) Electronic.—Normally, electronic selection is desired and is, indeed, necessary for high-speed devices.
13. Machine limitations: (a) Number of rectifiers; (b) Number of tubes; (c) Mechanical complexity; (d) Cost; (e) Unique features.
14. Applications.—The breadth of application of the technique is, of course, important.
15. Required type of pulse input from information source.—The technique should be capable of making use of general input rather than be limited to outputs of specific devices or media, such as punched tape.

Another result of the study indicated that the most universal technique, if such exists, consistent with the above criteria, would be an electrostatic technique in which the actual recording, as such, is accomplished without moving parts. In addition, basic to a concept evolved from the study was that the recording should be done while the paper is moving and that alphameric symbols should be formed from a matrix array of dots. Continuously moving recording paper and matrix arrays tend towards economical embodiments, particularly for designs aimed at the higher printing speeds. Incidentally, the matrix character formation also makes direct character recognition more feasible.

An electrostatic process has at least several basic inherent advantages:

1. High-speed capabilities.
2. Low power dissipation.
3. High physical forces in the electric fields.

Discussion of the Technique

The electrographic recording technique appears to satisfy the criteria set forth very favorably. The features of the technique are listed as follows:

1. It is a high-speed technique; a mark can be put on paper in a duration as small as one microsecond, and printed characters formed from a 5 by 7 matrix can be recorded at rates exceeding 5,000 characters per second. For instance, recording rates at hundreds of inches per second of paper strip is potentially possible.
2. The only motion involved in the technique is the continuously moving paper under the printing head.
3. The system is basically low in cost.
4. The system consumes very little power; the power required to print 5,000 characters per second, serially, is about 5 watts in addition to that necessary to move the paper and fix the recorded images.
5. The system is relatively quiet; the only noise is due to the moving paper.
6. The printing technique does not involve any messy, wet, or damp processes.
7. Permanent recording with no fading is achieved.
8. No wear of electrodes or the like is involved.

The electrographic recording technique produces controlled, visible dots by electrical pulse means directly. In its essentials, the process utilizes a controlled source of charge to form small charged areas on a high-resistivity surface such as a coated paper. The electrostatic latent image formed by the charged areas is made visible by inking with a single suitable powder and made permanent by thermal fixing. In applications in which the images are to be erased and the medium reused, the thermal fixing stage is eliminated. Fig. 1 shows the rudiments of the technique. During the recording stage the electrical discharge from the point electrode to a grounded metal plate is used as the source of charge to form the electrostatic latent image on the high-resistivity paper surface. The size and shape of the image depend mainly upon the polarity, the electric field strength and the surface coating used on the paper. A relatively low negative voltage applied to the point electrode gives small round dots suitable, for instance, for high-speed matrix printing. The recording medium is a relatively low-cost, uniformly and smoothly coated paper. The coating is a colorless, high-resistivity, thermoplastic coating. The thermoplastic feature of the coating in combination with a suitable ink and appropriate heat processing makes it possible to make the developed electrographic image completely permanent. The electrographic ink consists of a single powder consisting of material colored as desired. To ink the latent image the paper is passed through an inker containing the powder to give a visible image with virtually no background discoloration. The image is made permanently visible by passing the inked paper over a temperature-controlled hot plate. The three steps in the recording process are necessarily consecutive and are performed as the paper moves continuously at the appropriate speed for the particular recording application.

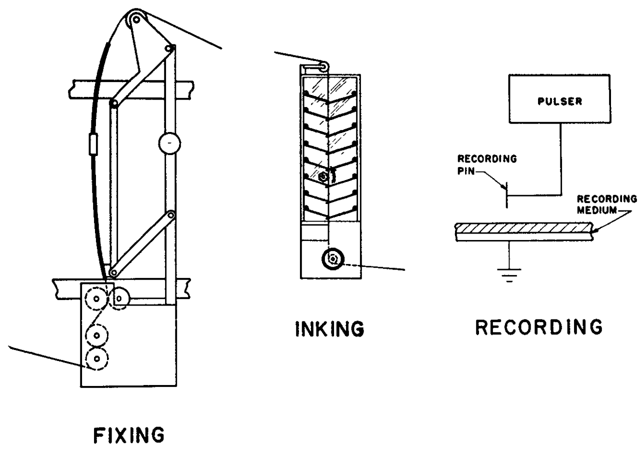


Fig. 1—Rudiments of electrographic printing.

Fig. 2 illustrates the recording head structure and the 5 by 7 matrix character formation. The recording head in this case is a set of seven 0.005-inch diameter flat-faced wires in a row, maintained at a fixed distance from the surface of the continuously moving paper. The figure shows sketches of the head. The character is built up by five successive choices of the seven pins. Fig. 3 is the electronic pulsing circuit; one is used for each pin. The circuit consists of one standard tube transformer coupled to the printing pin. A 250-volt supply voltage is used with an input voltage of the proper duration and amplitude; for instance, 40 microseconds and 20 volts. A voltage bias below the recording threshold is maintained at the printing pins. This allows the use of smaller pulses and closer pin spacing in the head.

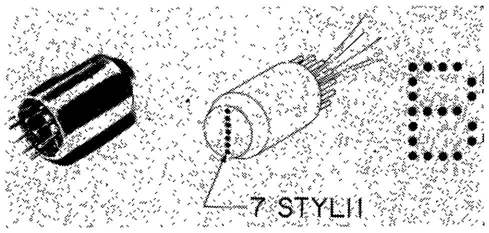


Fig. 2—Recording head structure.

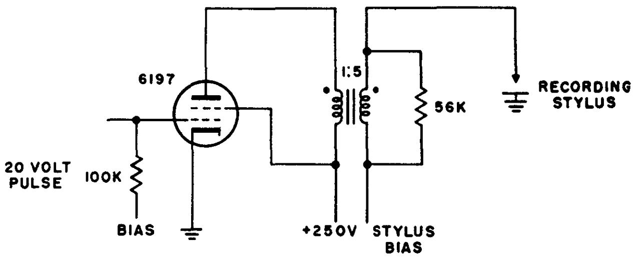


Fig. 3—Electronic pulsing circuit.

There are other features of the technique which may be of interest in special applications:

1. The latent electrostatic image and inked visible image will remain on the paper for a long time before fixing.
2. The powdered ink and corresponding electrostatic image can be erased, if desired, and new information recorded on the same medium. Of course, at any time, the inked image on the medium can be made permanent by carrying through the heat fixing.
3. It is possible to detect latent image electrically before inking. Thus, the technique offers possibilities for use of the recorded medium after each stage of the process; that is, recording, inking and fixing.

Fig. 4 shows a 5 by 7 magnetic matrix and decoding unit to buffer the pulsing circuits to the pins and the information source. The high pulsing rates made possible by this printing technique allow reasonably high recording speeds to be obtained with seven pins in five pulse times, using one decoding matrix and one 5 by 7 magnetic matrix. This results in a small-size, low-powered, cheap recording device. If straight parallel printing with a buffer unit for each channel is used, then phenomenally high speeds are made possible. In fact, the limitation becomes the speed at which paper can be handled.

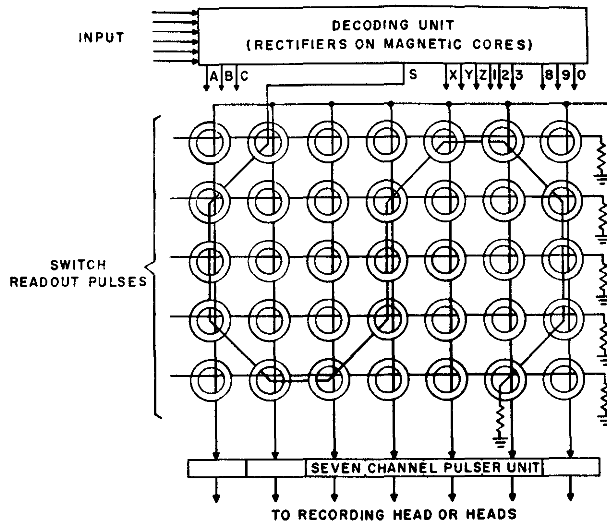

INPUT
DECODING UNIT (RECTIFIERS ON MAGNETIC CORES)
A B C
S
X Y Z 1 2 3
8 9 0
SWITCH READOUT PULSES
SEVEN CHANNEL PULSER UNIT
TO RECORDING HEAD OR HEADS

Fig. 4—Decoding unit.

Applications

Fig. 5 is a picture of the laboratory model which has been used in the investigations of the technique. Some other applications of the particular technique are as follows:

1. High-speed labeling or strip-printer applications.
2. Digital computer output systems.
3. Page-printer applications including plotters.
4. Teletyping and telemetering applications.
5. High-speed strip-chart type of recorder.
6. Facsimile applications.
7. Applications in which the capabilities of high-speed recording coupled with high-speed reading (photoelectric or magnetic reading) are necessary, made possible through the use of a dark-colored magnetic inking powder in the technique.

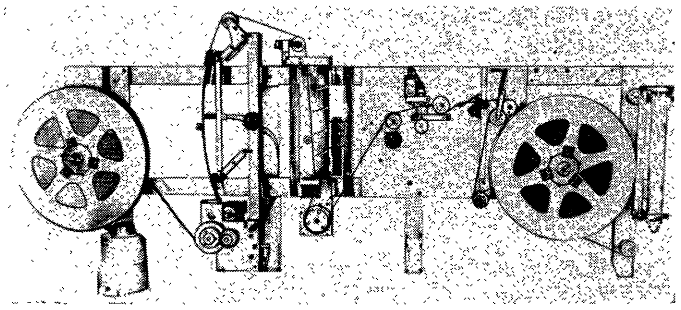


Fig. 5—Laboratory model of electrographic printer.

Acknowledgment

The author would like to acknowledge the contribution to the development of the subject technique by F. Innes, R. Phelps, and O. Stram.

An Electronic Digital Polynomial Root Extractor

R. R. JOHNSON†

Summary—**Many mathematical techniques exist for factoring algebraic polynomials. Most require much computation and programming and are practical only for large machine computers. A different approach to the general problem is described. The mathematical method is an adaptation of a Taylor series approximation used to connect the problem and its formulation with a special machine implementation. The result is a simple digital computer capable of extracting the complex roots of an *n*th degree polynomial.**

Introduction

THE ADVENT OF the large-scale digital computer has resulted in a general emphasis on numerical methods. This emphasis has led to many applications of digital techniques to problems having special characteristics; computers designed to capitalize on these characteristics can obtain solutions rapidly and with considerable savings in computing equipment. Reductions in preparation and programming time can result from machines designed to handle problems appearing repeatedly in practice.

In scientific computations, one universally belabored problem is that of factoring algebraic polynomials.[1–3] Many analog devices[4–10] and many mathematical methods[11–18] have been developed to solve this problem. A special-purpose computer has been designed and constructed at the California Institute of Technology to reduce the programming and scheduling delays involved in placing polynomials in large-scale machines. Other reasons for its construction were the development of new computer techniques and the desire for a machine having more versatility and accuracy than any of the devices now available.

This computer is designed to handle polynomials up to the sixteenth degree. Operating in the binary number system, it requires 20 flip-flops and approximately 200 germanium diodes. The operating memory is one circulating register with one clock channel on a small magnetic drum. Input is bit by bit, using a pair of switches, and output is visual on an oscilloscope. The only modifications necessary to extend this to higher degree polynomials would be a larger drum or a higher pulse density. The computer obtains the complex roots of polynomials having real or complex coefficients with an accuracy of approximately six decimal digits. Solution times average about 16 seconds per root.

Principle of Operation

The general method is that of evaluating a polynomial in the complex domain. The computer is designed to: (1) Evaluate the polynomial for successive increments in its complex argument, and (2) select the complex increment that always decreases the absolute value of the polynomial. A brief description is given of the problem preparation requirements and of the numerical accuracies attainable.

Mathematical Techniques

The computer generates values of the polynomial by repeated steps of Δ in its complex argument. Before each step the value

$$\pm\delta = \Delta$$

$$\pm j\delta = \Delta$$

is chosen such that the step will diminish the absolute value of the polynomial. In this fashion the argument is modified as the computer assumes the direct path to the

† California Institute of Technology, Pasadena, Calif.

[1] E. T. Whittaker and G. Robinson, "The Calculus of Observations," Blackie and Son, London, ch. VI; 1940.

[2] Fr. A. Willers, "Practical Analysis," Dover Press, New York, ch. IV; 1948.

[3] A. S. Householder, "Principles of Numerical Analysis," McGraw-Hill Book Co., Inc., New York, ch. 3; 1953.

[4] S. L. Brown and L. L. Wheeler, "Mechanical methods for graphical solutions of polynomials," *Jour. Franklin Inst.*, vol. 231, pp. 223–243; March, 1941.

[5] B. O. Marshall, Jr., "Electronic isograph for roots of polynomials," *Jour. Appl. Phys.*, vol. 21, pp. 307–312; April, 1950.

[6] E. O. Willoghby, G. A. Rose, and W. G. Forte, "Analog Computers to Solve Polynomial Equations with Real Coefficients," Proc. Conference on Automatic Computing Machinery, Commonwealth Scientific and Industrial Research Organization August, 1951.

[7] D. Herr, "Two new economical computers for design and analysis of servomechanisms, networks, amplifiers, and analogous dynamical systems," *Amer. Soc. Nav. Eng. Jour.*, vol. 63, pp. 950–971; November, 1951.

[8] F. W. Bubb, Jr., "Circuit for generating polynomials and finding their zeros," PROC. IRE, vol. 39, pp. 1556–1561; December, 1951.

[9] L. Löfgren, "Analog computer for roots of algebraic equations," PROC. IRE, vol. 41, pp. 907–913; July, 1953.

[10] M. L. Morgan, "A Computer for Algebraic Functions of a Complex Variable," Ph.D. thesis, Calif. Inst. Tech.; 1954.

[11] S. N. Lin, "A method of successive approximations of evaluating the real and complex roots of cubic and higher order equations," *Jour. Math. Phys.*, vol. 20, pp. 231–242; January, 1941.

[12] P. A. Samuelson, "Iterative computation of complex roots," *Jour. Math. Phys.*, vol. 28, pp. 259–267; January, 1949.

[13] Y. L. Luke and D. Ufford, "On the roots of algebraic equations," *Jour. Math. Phys.*, vol. 30, pp. 94–101; January, 1951.

[14] D. B. Steinman, "Simple formula solves all higher degree equations," *Civil Engrg.* (N. Y.), vol. 21, pp. 44–45; February, 1951.

[15] F. W. J. Oliver, "Evaluation of zeros of high degree polynomials," *Phil. Trans. Roy. Soc. London*, vol. 244, pp. 385–415; April, 1952.

[16] H. E. Salzer, "Calculating zeros of polynomials by method of Lucas," *Res. Jour.*, U. S. Bureau of Standards (RP 2348), vol. 49, pp. 133–134; August, 1952.

[17] J. F. Koenig, "On zeros of polynomials and degree of stability of linear systems," *Jour. Appl. Phys.*, vol. 24, pp. 476–482; April, 1953.

[18] L. Tasny-Tschiassny, "Location of roots of polynomial equations by repeated evaluation of linear forms," *Quart. Appl. Math.*, vol. II, pp. 319–326; October, 1953

nearest root. When it reaches a minimum value for the polynomial, the normal Δ selection causes the computer to encircle the point of minimum absolute value.

The single memory channel contains 19 word positions. The real and imaginary components of the polynomial are in the first word; in each of the following 16 words is located the corresponding derivative of the polynomial (see Fig. 1).

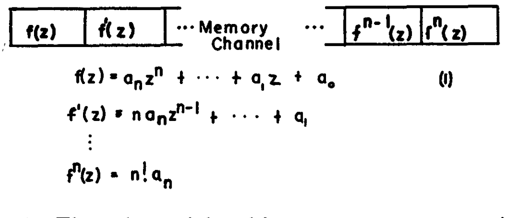

f(z)
f'(z)
Memory Channel
f^{n-1}(z)
f^n(z)
f(z) = a_n z^n + ··· + a_1 z + a_0 (1)
f'(z) = n a_n z^{n-1} + ··· + a_1
f^n(z) = n! a_n

Fig. 1—The polynomial and its computer representation.

These derivatives, evaluated at some convenient point such as the origin, comprise the initial input data. The computational principle is to evaluate each derivative at an incremental distance Δ away from the initial co-ordinate (see Fig. 2). This approximation is the first term of a Taylor series expansion of the polynomial. All derivatives are recomputed for each step Δ using this principle.

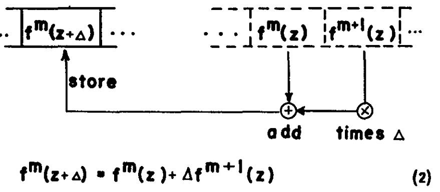


Fig. 2—The computation principle.

For the initial root location an improved approximation formula is used with a larger value of δ. Higher-order terms could be taken from the Taylor series, but a more elegant technique is to use part of each newly computed derivative:

$$f(z+\Delta) = f(z) + \tfrac{1}{2}\Delta f'(z) + \tfrac{1}{2}\Delta f'(z+\Delta). \quad (3)$$

The $(m-1)^{st}$ derivative is computed:

$$f^{(m-1)}(z+\Delta) = f^{(m-1)}(z) + \tfrac{1}{2}\Delta f^{(m)}(z) + \tfrac{1}{2}\Delta f^{(m)}(z+\Delta). \quad (4)$$

Repeated approximations permit the computer to evaluate its polynomial for any complex argument. Coupled with the direction decision elements, the computer follows the shortest path to the nearest zero of the polynomial.

Several alternatives exist for finding all n roots. The method used is to continue the normal computation cycle but to force a desired Δ selection. The operator chooses the direction in which he wishes to look for roots and forces the computer to move in that direction. Released, the computer seeks the nearest root. Complex conjugate roots can be eliminated immediately.

Direction Decision

Consider the simplified approximation formula; the real and imaginary components for each step are found:

$$f(z) = u(z) + jv(z).$$

For $\Delta = \pm\delta$,

$$\begin{aligned} u(z+\Delta) &= u(z) \pm \delta u'(z), \\ v(z+\Delta) &= v(z) \pm \delta v'(z). \end{aligned} \quad (5a)$$

For $\Delta = \pm j\delta$,

$$\begin{aligned} u(z+\Delta) &= u(z) \mp \delta v'(z), \\ v(z+\Delta) &= v(z) \pm \delta u'(z). \end{aligned} \quad (5b)$$

The selection of Δ is that which will assure

$$|f(z+\Delta)| \leqq |f(z)|.$$

For example, if both u and v are positive and their first derivatives are positive, $\Delta = -\delta$. However, if u and v are positive and their first derivatives differ in sign, $\Delta = \pm j\delta$ is the proper selection.

A simple case illustrates the behavior of the direction control when z approximates a root to within an amount δ. Consider all signs initially plus. The sequence of decisions prescribed by the rules above forces the computer to encircle the root. The computer remains in this mode until forced in another direction.

Problem Preparation

There are two restrictions placed on the polynomial. It is necessary to convert the coefficients to binary numbers and to limit all numbers to absolute values less than 1.0. There are 30 significant binary places available for the real and imaginary components of each derivative. Numbers are absolute value with sign when positive and zero's complements with sign when negative (see Fig. 3).

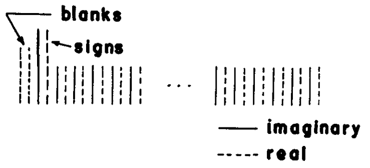


Fig. 3—The biplexed word structure for each derivative.

To insure that $|z| \leqq 1$ for all roots, it is necessary to compute the radius of the contour in the z plane which encloses all n zeros of the polynomial

$$R > \left|\frac{a_k}{a_n}\right| - 1, \quad (6)$$

where $|a_k|$ is the largest coefficient in the polynomial. A new argument is defined:

$$w = \frac{z}{R}, \quad (7)$$

which transforms the polynomial into

$$f(w) = a_n(R)^n w^n + \cdots + a_1(R)w + a_0. \quad (8)$$

The computer obtains the roots of (8). The roots of the original polynomial are found using (7). To insure that the absolute value of the n derivatives never exceed 1.0, it is necessary to divide $f(w)$ by the largest coefficient:

$$F(w) = \frac{f(w)}{|2n!R^i a_i|}, \tag{9}$$

$$= A_n w^n + \cdots + A_1 w + A_0, \tag{10}$$

where i is the index of the largest coefficient in (8).

These computations must be done manually before inserting the derivatives into the computer. This is done to retain the computational simplicity of the computer. No multiplications or divisions are provided internally.

D. Accuracy

Each step in the approximation of the function and its derivatives has a truncation error

$$\epsilon = \sum_{i=m+3}^{n} \frac{\Delta^i}{2i!}(i-2)f^{(i)}(z). \tag{11}$$

This is the error in the approximation to the mth derivative. The error in the function itself is of the order of

$$\frac{\Delta^3}{12} f^{(3)}(z).$$

Two values of δ are used: 2^{-10} and 2^{-20}. The computation sequence is to locate the root with the larger and refine it with the smaller δ. With the coarse δ, the truncation error at each step is of the order of 2^{-34}. The total error in 2^{10} steps is 2^{-24}. The root is refined using approximation (2) with $\delta = 2^{-20}$. In 2^{10} steps this produces an error of 2^{-32}.

Roundoff errors may propagate to the twenty-first binary position in 2^{10} steps so that the polynomial, its derivatives, and the value of z may be considered accurate to the twentieth binary place. The smallest increments in z are $\delta = 2^{-20}$.

To obtain full accuracy for all roots, it is necessary to normalize by (7) and locate the roots approximately. Full significance is obtained for the smaller roots by renormalizing the polynomial to an R just greater than the modulus of the desired root.

Machine Design

The computer has three modes of operation:

1. Input.
2. Computation with $\delta = 2^{-10}$ or $\delta = 2^{-20}$,
3. Direction decision.

The operational aspects of these modes have been described. This section gives a general description of the techniques by which that functional behavior is accomplished. The detailed logical design will be available from the California Institute of Technology.

Input

To describe the input routine, it is necessary to understand some of the internal features of the computer. Input itself is so simple that it will be convenient to present a more detailed picture of the computer in this section as well.

During the first moments of the input phase the computer orients itself with respect to its internal timing. This feature arises from a general consideration of the computer configuration (see Fig. 4).

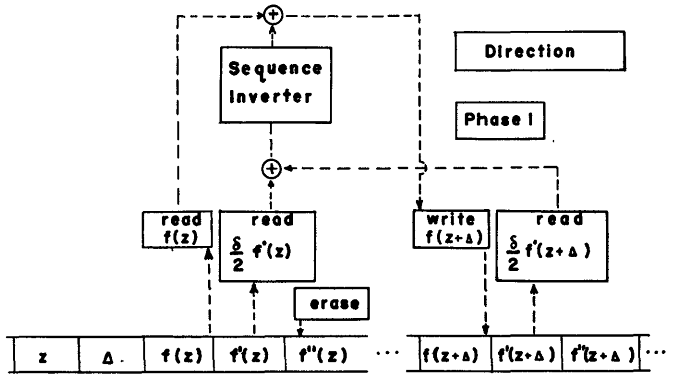


Fig. 4—The computer block diagram.

To minimize the number of reading and writing heads on the drum, it was necessary to use one circulating register. This implied a single clock channel with no reference to an origin pulse on the drum. Timing signals are required, however, to indicate certain word positions; without having the circulating register exactly one drum circumference in length it is necessary to generate certain timing markers inside the register itself.

In addition to the polynomial and its derivatives, the memory channel contains two words, "z" and "Δ." "Δ" holds a marker used to add the correct Δ into "z." "z" holds the value of the complex argument z of the polynomial. The timing signals are those necessary to distinguish:

1. The words $f(z)$, "Δ," "z," and $f^{(n)}(z)$,
2. The internal word structure: blank bits, sign bits, real or imaginary bits, and the position of the δth binary place.

Phase Control designates the first group of timing signals; the second group are general timing signals.

The blank bits between words may contain two markers. These markers are entered under the control of the general timing signals when the input mode switch is closed. One marker precedes the words holding $f(z)$ and "z"; it serves to synchronize the phase control. The other is a movable marker identifying any particular word.

The general timing signals originate in the clock channel. Originally the drum is uniformly divided into 1,344 positions. The clock channel is derived by occasionally omitting a single pulse (see Fig. 5). The clock pulses to operate the computer are obtained from a free-running phantastron synchronized by pulses from the clock channel. An omitted pulse is filled in for clock-pulse use by the phantastron. With one pulse omitted at a time, there is little drift. The missing pulse operates the general timing flip-flops; they change state whenever a pulse is omitted from the clock channel. Synchronization of the general timing pattern with the drum pattern is self-controlled and may take several word times after the computer is turned on.

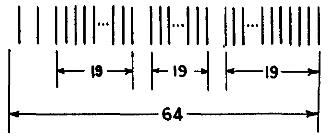


Fig. 5—One word of the pulse configuration on the clock channel.

The input bits are entered via two lever switches; each bit is inserted in the word identified by the movable marker. Input is most significant bit first. The bit is placed in a flip-flop and the desired word circulates back to the memory through that flip-flop.

Computation

There are three operations in the computation mode:
1. Multiplication by $\frac{1}{2}\delta$,
2. Inversion of the bit sequences as indicated by (5),
3. Addition of the respective components.

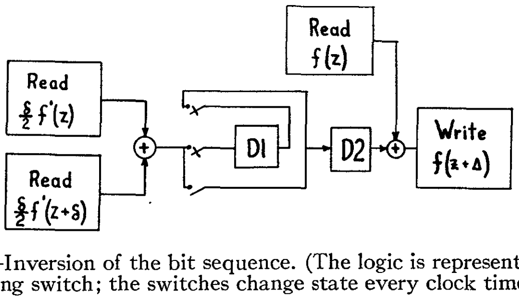
Multiplication by $\frac{1}{2}\delta$ is done by shifting the reading heads. This is indicated in Fig. 4; it is also the reason for limiting the absolute values to 1.0.

Inversion merely exchanges the biplexed positions of the real and imaginary components when Δ changes from real to imaginary. Two flip-flops are used for the inversion, and two separate adders are used to obtain a compromise between numbers of flip-flops and numbers of diodes.

Subtraction or addition is governed by Direction Control. $\frac{1}{2}\delta f'(z)$ and $\frac{1}{2}\delta f'(z+\Delta)$ are always added, but their sum is added to or subtracted from $f(z)$. The simplest technique is to combine the conversion to zero's complements of the subtrahend with the sequence inversion.

Direction Control

This is the most complicated logical operation in the computer. The decision is based on a truth table of all sign conditions in (5). The logic is derived from a similar table based on the information available internally in the computer. That is, due to sequence inversion and zero's complementation in D1 and D2 (Fig. 6), it is



Read $f(z)$
Read $\frac{\delta}{2}f'(z)$
Read $\frac{\delta}{2}f'(z+\delta)$
D1
D2
Write $f(z+\Delta)$

Fig. 6—Inversion of the bit sequence. (The logic is represented by a gang switch; the switches change state every clock time.)

necessary to consider the form in which the sign information is available. One additional limitation on the decision logic is that the modified sign bits are not all available during any one clock time.

Two flip-flops, G and H, are used to perform the decision. After each circulation of the memory channel a new value for Δ is determined from the new signs of $f(z)$ and $f'(z)$ and from the original states of G and H. Referring to Fig. 6 and using the additional notation

$B \equiv$ odd-even clock timer,
$M1 \equiv$ reading flip-flop for $f(z)$,
$T0 \equiv$ sign bit times (the real sign is in $M1$ during $\overline{B}\cdot T0=1$, and the imaginary sign is in $M1$ during $B\cdot T0=1$),

the input equations to G and H are written:

$$\begin{aligned} 1G &= (D1 \times D2 \times B)\cdot\overline{(D2 \times M1)}\cdot T0 \\ 0G &= \overline{1G} \\ 1H &= \overline{(D2 \times M1)}\cdot T0 \\ 0H &= 1H. \end{aligned} \tag{12}$$

These equations are given to illustrate the form in which the logical design is obtained. The bar denotes complementation and the ($\times$) denotes the exclusive "or" operator

$$D1 \times D2 = D1\cdot\overline{D2} + \overline{D1}\cdot D2.$$

The $+$ and the $\cdot$ symbols designate logical union and intersection, respectively. Eqs. (12) describe a flip-flop having the following behavior.

Input:		Output:	
$1G_n$	$0G_n$	G_{n+1}	
0	0	G_n	
0	1	0	(13)
1	0	1	
1	1	$\overline{G_n}$	

The output is delayed one clock interval with respect to the input; n denotes the clock interval.

The decisions produced by (12) are not exactly correct. In a few cases G and H will select the correct real or imaginary Δ but will give it the wrong algebraic sign. The computer takes one step in the wrong direction and then recognizes its mistake and proceeds properly.

This is a situation where it is easiest to make an occasional wrong decision without hurting the functional operation of the device. Several additional diodes and an extra flip-flop would be required to make the correct decision at all times.

Physical Specifications

Two types of flip-flops are used. Those described by (13) are the first. The second kind are described by the middle two rows of (13). The second type introduces a delay and power amplification only (see Fig. 7).

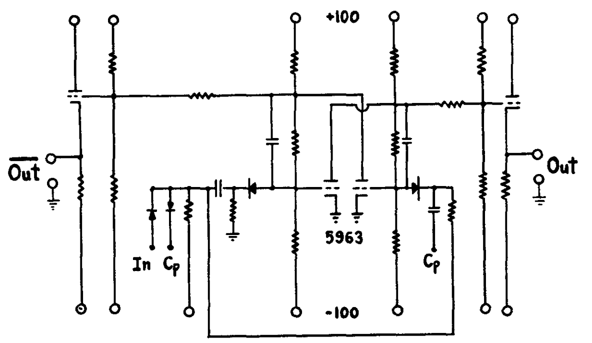


Fig. 7—Single-Input Flip-Flop. (The output swings from +5 to +20 volts; the clock pulse is referenced to +20 volts.)

Note that only two power supplies are required. These supply the flip-flops, diode matrices, and the read-write amplifiers. Approximately 100 watts is dissipated exclusive of heaters.

The basic clock frequency is 80.7 kc derived from 1,344 pulses on a 5-inch drum rotating at 3,600 rpm.

Mode selection is under the control of several switches on the control panel. They are used for starting, inserting the timing markers, input, Δ selection, forced direction control, locating the movable marker, and selecting any desired output (see Fig. 8).

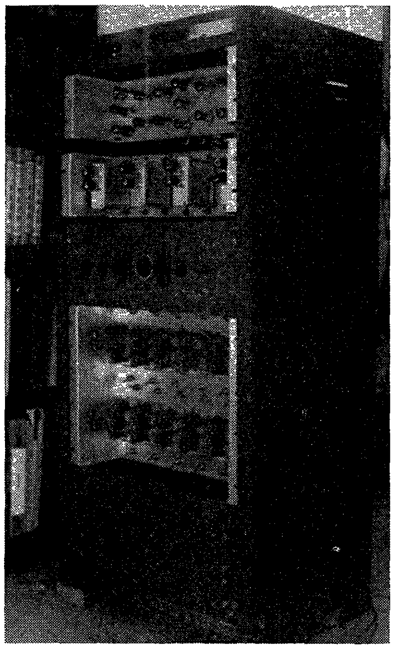


Fig. 8—The computer.

Conclusion

Several observations result from the design and construction of the machine. It requires about half the equipment necessary for a medium-speed general-purpose computer of moderate memory capacity. Where there is need for its special purpose, the ready availability of the computer and its ease of programming make it a valuable scientific tool. The binary coding and bit by bit input and output are limitations, but it is still felt that the equipment savings warrant them. Even more equipment could be eliminated if certain other features of the present computer were omitted.

While the logical techniques developed for the root extractor are conceptually useful in other digital devices, the computer itself has several other uses.

The machine can be considered as a function generator in two variables in the sense that it evaluates a polynomial of high degree. Inputs are any desired x and y. Internal programming is set to approach the x and y inputs rather than to diminish $|f(z+\Delta)|$. Discontinuous functions could be generated if there were some particular behavior characteristic of a higher derivative. Input could be to the derivative word position in this case.

In the field of servomechanisms the computer could be used to obtain root locus plots. A curve-plotter would be attached to read the "z" word in this case, and some form of internal gating would be used to indicate that a root had been located.

Acknowledgment

The material described above covers a portion of the work performed at the California Institute of Technology by the author in partial fulfillment of the requirements for the Ph.D. degree in Electrical Engineering. It was done during periods when the author was the holder of a Howard Hughes Fellowship and an International Business Machines Corporation Fellowship in Electrical Engineering. Particular acknowledgment is due Dr. Stanley P. Frankel for his invaluable ideas and suggestions contributed during the development of the computer, and to Dr. G. D. McCann for his helpful assistance throughout the project's duration.

A Set of Transistor Circuits for Asynchronous, Direct-Coupled Computers

R. A. KUDLICH†

EXTENSIVE operating experience with the ORDVAC[1] and the ILLIAC has demonstrated to members of the University of Illinois Computer Laboratory the advantages of computers which are direct-coupled and operate asynchronously. This paper describes a set of building block circuits, which is the first result of a program to develop transistor circuits which would be suitable for an asynchronous machine.

This paper will not go into the detailed mathematics of the circuits. Instead, its purpose will be to explain the design procedure that was used, and to describe briefly some of the results obtained with the circuits. A relatively complete discussion of the design of the flip-flop circuit will be used to illustrate the general technique, whose principle is undoubtedly familiar to many readers. Instead of studying the behavior of the circuit for the nominal values of the components, expressions are obtained for its behavior when every parameter value varies to some specified extent. The circuit is designed to operate properly with these extreme values; ordinarily a malfunction will then occur only if several components vary simultaneously by slightly more than the allowable amounts, or if a single parameter exceeds the allowable variation to a very great extent. (One example of the latter might be the sudden decrease of the current gain to zero when a point contact transistor burns out.)

Two fundamental characteristics of asynchronous computers are vital considerations in the design of such circuits. First, the binary signals in a direct-coupled computer are dc voltage levels rather than trains of pulses produced by a clock. Second, the individual switching operations, such as changing the state of a flip-flop, do not have any predetermined maximum duration. Instead, the machine waits until a signal has been produced, signifying the successful completion of the operation, before initiating the succeeding step. As a result, variations in the transient response of the individual circuits will not cause a malfunction. Because of these two factors, it is feasible to concentrate upon dc behavior when designing building block circuits for asynchronous operation.

The flip-flop circuit is shown in Fig. 1. Two point contact transistors are used, with their emitters connected together and returned through a bias resistor, R_e', to a positive supply voltage, V_{ee}. To the best of the present writer's knowledge, this circuit was first proposed at the Air Force Cambridge Research Center.[2] This circuit can have two stable states, in each of which one transistor is cut off, the other conducting.

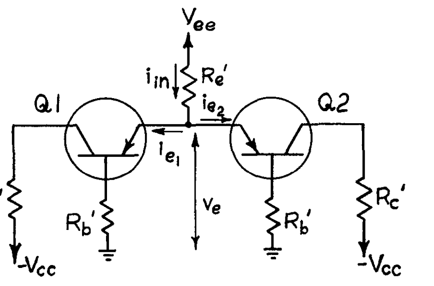


Fig. 1—Flip-flop circuit with two point-contact transistors.

With appropriate values for R_b' and R_c', the relation between v_e, the emitter-ground voltage, and i_e, the emitter current for either transistor, is the familiar N-curve, shown in Fig. 2. The N-curve represents an approximation to the nonlinear characteristics of the transistor in which the device parameters are considered to be constant throughout each of three operating regions.[3] These regions are:

I. Cutoff Region: emitter current is zero or negative.
II. Active Region: normal transistor action takes place.
III. Saturation Region: collector-base voltage is too small to maintain transistor action.

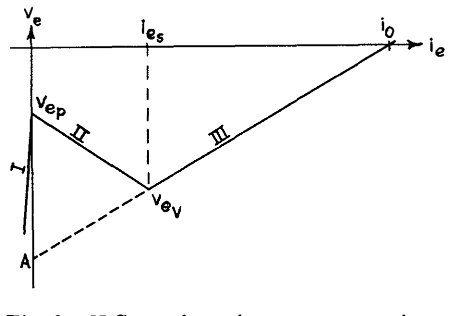


Fig. 2—N-Curve for point-contact transistor.

Values for the N-curve which are critical for the design of the flip-flop circuit are the peak voltage, v_{e_p}, the

† Bell Telephone Laboratories, Whippany, N. J.; formerly University of Illinois, Urbana, Ill.

[1] R. E. Meagher and J. P. Nash, "The ORDVAC," *Rev. of Elec. Digital Computers*, pp. 37–43; 1952.

[2] A. W. Carlson, "High Speed Transistor Flip-Flops," AFCRC Tech. Report 53-16; June, 1953.

[3] A. E. Anderson, "Some switching aspects of transistors," *The Transistor, Bell Telephone Labs*, pp. 283–334; 1951.

saturation current, i_{e_s}, and the valley voltage, v_{e_v}. Both the zero-voltage current, i_0, and the saturation-region slope, r_{III}, are almost independent of the transistor characteristics. The approximate expressions which are used for these values are listed below:

Approximate expressions used in the analysis:

1. $v_{e_p} = -R_bV_{cc}/R_b + r_c + R_c'$
   $(R_b = R_b' + r_b)$.
2. $i_{es} = V_{cc}/(a-1)R_b + aR_c'$.
3. $v_{e_v} = -(a-1)R_bV_{cc}/(a-1)R_b + aR_c'$.
4. $i_0 = V_{cc}/R_c'$.
5. $r_{\text{III}} = R_bR_c'/R_b + R_c'$.

The expressions given imply the following abrupt changes in parameter values:

1. The current gain[4] a changes from zero to its active-region value at zero emitter current.
2. The collector resistance, r_c, is constant throughout regions I and II, and becomes zero throughout region III.

It can be seen that if variations in external components and r_b are neglected, then v_{e_p} is a function only of r_c, i_{e_s} and v_{e_v} are functions only of a, and that i_0 and r_{III} are constants.

This is taken into account in Fig. 3, which shows two *N*-curves (dashed lines) for transistors with different parameter values, and segments of the combined input characteristic for both units, as seen at the common-emitter point of Fig. 1. (Note that the region III segment is the same for the two *N*-curves.) If transistor *Q*1 has the minimum permissible r_c value, and *Q*2 the maximum, then all acceptable transistors will have peak voltage values between those of *Q*1 and *Q*2. Similarly, if

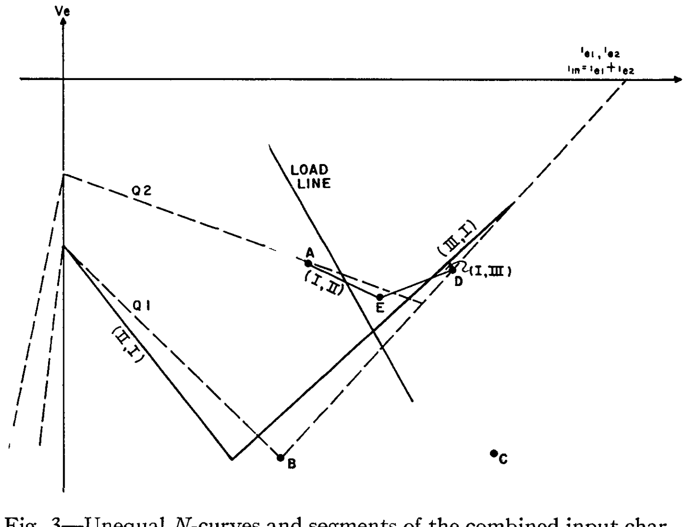


Fig. 3—Unequal *N*-curves and segments of the combined input characteristic ($v_e - i_{in}$) for the circuit of Fig. 1. (Dashed lines are the *N*-curves.)

[4] The current gain a used here is the ratio r_m/r_c and for most transistors is almost equal to alpha, the short-circuit current gain.

*Q*1 has the maximum permissible value of a, and *Q*2 the minimum, valley points for all acceptable transistors will lie on the region III segment between those for *Q*1 and *Q*2. A flip-flop circuit which operates correctly for all combinations of these extreme peak and valley voltages will therefore function properly with any pair of acceptable transistors.

The solid lines indicate the relation between v_e, the common-emitter voltage, and i_{in}, the total current flowing to both transistors through R_e'. The various segments correspond to different combinations of operating conditions for the two transistors. For example, the segment marked (II, I) represents the combined input characteristic when *Q*1 is operating in the active region and *Q*2 is cut off.

If the circuit is to be bistable, it is necessary that the peak point for *Q*1 be less negative than the valley point for *Q*2; otherwise, there will be no stable state with *Q*1 cut off and *Q*2 conducting. The first requirement to be met is:

$$|v_{e_v}|_{\text{min}} > |v_{e_p}|_{\text{max}}. \tag{1}$$

This inequality is set up in terms of the extreme allowable transistor parameters (e.g., minimum r_c in $|v_{e_p}|_{\text{max}}$), and of the specified tolerances for supply voltages and external resistors.

The additional requirements which were placed upon this particular circuit are those which allow greatest variation in the transistor characteristics. To begin with, it is necessary to preclude the possibility of simultaneous saturation of both transistors, since this would be a third stable state. Thus the load line for R_e' and V_{ee} must lie to the left of point *C* of Fig. 3, which corresponds to a total input current twice as large as the minimum saturation current (i_{e_s} for *Q*1, point *B*). The second expression is therefore:

$$\frac{V_{ee} + |v_{e_v}|_{\text{max}}}{R_e'} < 2(i_{e_s})\text{ min}. \tag{2}$$

The left-hand side of (2) is the available current through R_e'.

Faster operation could be obtained by insuring that the conducting transistor did not saturate, but operated in the active region in every instance. However, this would require that the load line lie somewhat to the left of point *B*, and thus further restrict the choice of R_e' and V_{ee}.

The third condition imposed to insure bistable operation is that the load line must be to the right of point *A*, in order to have a stable state with *Q*2 conducting, *Q*1 cut off. Requirement (1) insures the possibility of this state; the present requirement insures a load line position which utilizes this state. (For reasonably large maximum a, and reasonably small minimum r_c, requirement (2) will prevent the load line being to the right of point *D*, Fig. 3.) The voltage at point *A* is $|v_{e_p}|_{\text{max}}$; the current is simply the difference between the peak points of *Q*1 and *Q*2, divided by the active-region slope for *Q*2.

The inequality to be satisfied is:

$$\frac{V_{ee} + |v_{e_p}|_{\max}}{R_e'} > \frac{|v_{e_p}|_{\max} - |v_{e_p}|_{\min}}{|r_{II}|_{Q2}}. \tag{3}$$

When requirements (1) through (3) are written in terms of transistor parameters and component values, the supply voltages V_{cc} and V_{ee} enter only as a ratio V_{cc}/V_{ee}.

Since some transistors will operate in the active region when conducting (e.g., $Q2$), it is necessary to limit the capacitance between the common-emitter point and ground to prevent instability and oscillation.[5] If the conducting unit were required to saturate in every instance, this limitation would not be imposed, but the load line would have to lie to the right of point E. In summary, allowing the conducting transistor to operate in the active or saturation region, depending upon its particular parameter values, has certain disadvantages but permits the greatest variation in the transistors themselves.

Having chosen acceptable ranges for the transistor parameters, based upon measurements or published data, expressions (1) through (3) can be used to find satisfactory values for the resistors and the ratio of the supply voltages. There is, of course, no certainty that all three requirements can be satisfied for any particular set of transistor specifications; as more stringent restrictions are placed upon the transistors, it becomes easier to satisfy the requirements.

The design of the remaining circuits followed the three-step procedure illustrated above:

1. The requirements which the circuit must meet are determined by careful study.
2. Mathematical conditions necessary to meet the requirements are derived.
3. Component values and device specifications are chosen to meet the mathematical conditions.

Consider now the voltage levels appearing at the collectors of the point contact transistors. Once the component values for the circuit have been chosen, the extreme values of all parameters are known, and the minimum and maximum collector voltages can be determined for both the conducting and the nonconducting unit. Fig. 4 shows the relation among various voltage levels; both ground and the collector supply voltage are shown as references. The shaded regions include the possible output voltages from the conducting transistor (v_1) and from the nonconducting transistor (v_0). The levels $-|v_0|_{\min}$ and $-|v_1|_{\max}$ are therefore the "worst" zero and 1 signals, respectively, produced by the flip-flop. However, as binary signals pass through successive stages of logic circuits, they may be attenuated, and the actual input signals to reach a given circuit may be "worse" than the extreme values just mentioned. Accordingly, two additional voltages, v_{zero} and v_{one} of Fig. 4, are designated the "defining values" for zero and 1 signals. These defining values include a margin for losses in the logic circuitry, and no acceptable signal in the computer will lie between these levels.

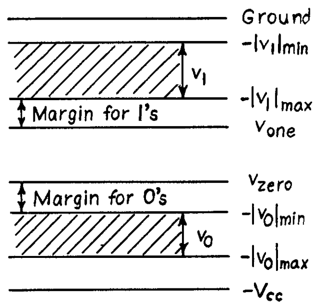


Fig. 4—Voltage levels of the Binary Signals. The range of collector voltage for the conducting flip-flop transistor is v_1; the range foı the nonconducting transistor is v_0.

The circuits other than the flip-flop are now designed to operate properly with input voltages of v_{zero} and v_{one}; they will then function satisfactorily for "better" binary signals. The levels $-|v_1|_{\min}$ and $-|v_0|_{\max}$ are important if the transistors in the other circuits are to be kept from saturating.

Associated with the flip-flop are read-out circuits and gating circuits; the combination will be termed a temporary storage device, abbreviated TSD. The complete TSD is shown in Fig. 5. $Q1$ and $Q2$ are the point contact transistors of the flip-flop circuit; the remaining transistors are junction units.

The read-out circuits are simply grounded-collector stages to provide impedance matching and isolation between the flip-flop and the logic circuits. The voltage drop across R_c' (or across the point contact transistor when *n-p-n* read-out transistors are used) insures that $Q3$ and $Q4$ will not saturate. At the complement output, a more positive signal is obtained when the flip-flop is in state zero, a more negative signal when the flip-flop is in state 1.

The flip-flop is set to the desired state by lowering the base voltage of the point contact transistor which is to be conducting (e.g., in Fig. 5 the base voltage of $Q2$ is lowered when the flip-flop is to be set to 1). The gating transistors, $Q5$ and $Q6$, which are normally nonconducting, do this by drawing current through R_b'. The clamping diodes prevent saturation of the gating units. (This type of clamping imposes an upper limit on alpha of the gating transistor to prevent excessive power dissipation.)

Once the flip-flop transistor begins to conduct, the common-emitter point will tend to follow any additional negative excursion of the base voltage, and the previously conducting point contact unit will be turned off. To minimize the turn-off time, the common-emitter point should be made at least as negative as point A of Fig. 2. Point A is the extension of the saturation-region segment of the N-Curve to the voltage axis. A transistor which has been saturated will tend to follow this extended characteristic, rather than the normal N-Curve,

[5] B. G. Farley, "Dynamics of transistor negative resistance circuits," PROC. IRE, vol. 40, pp. 1497–1508; November, 1952.

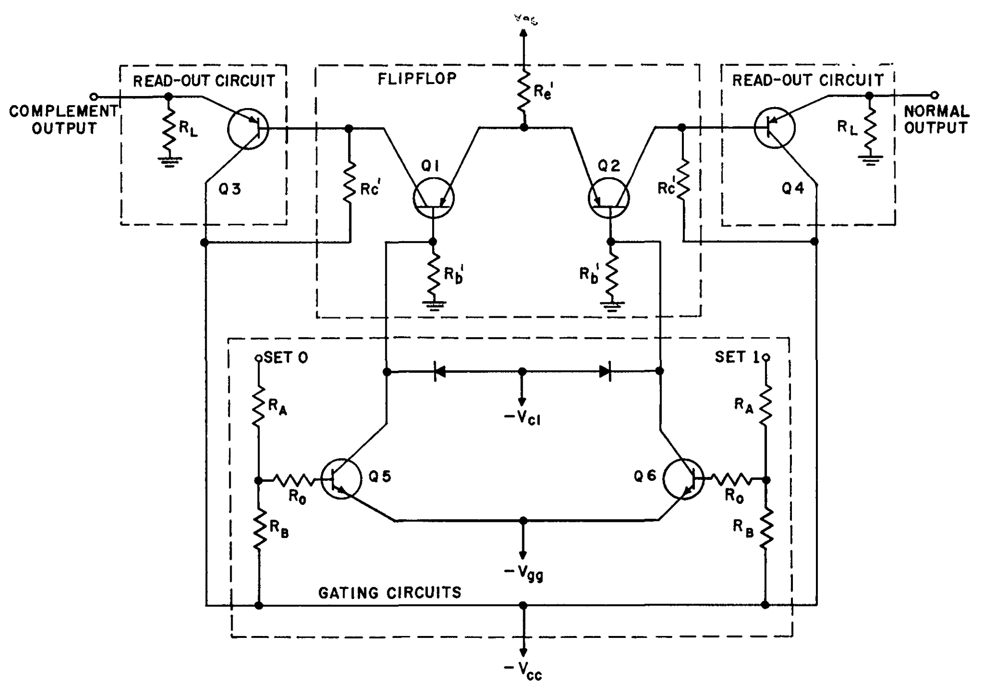


Fig. 5—Complete TSD circuit.

until the excess carriers have been removed. This voltage level is therefore necessary to reduce the emitter current to zero.

The design of the gating circuit follows the same technique as that of the flip-flop. The circuit requirements to be considered are that the gating transistor be cut off for zero input signals; that the flip-flop be set properly for 1 input signals; and that the gating transistor be prevented from saturating. Values of the resistors, supply voltages, and transistor parameters are chosen to meet these requirements.

The logic functions which are included in the building block circuits are "and," "or," and "not." Conventional diode circuits are used for the "and" and "or" functions, with the addition of a grounded-collector junction transistor circuit between successive stages.

Two "not" circuit designs are shown in Fig. 6. Circuit (*a*) is a conventional inverting amplifier, with a voltage divider circuit to restore the voltages at the collector to the normal 1 and zero signal levels. Circuit

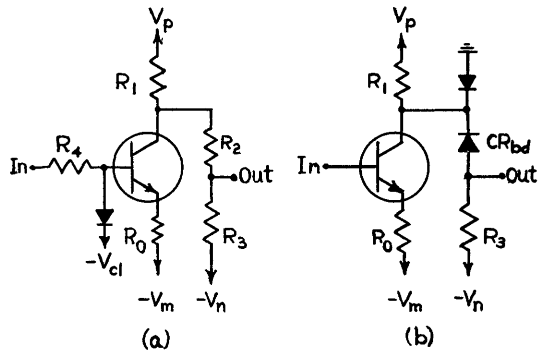


Fig. 6—Two types of "not" circuit with junction transistors. CR_{bd} is a silicon junction diode.

(*b*) uses a silicon junction diode for shifting the voltages. Because of its constant-voltage behavior at the breakdown voltage,[6] this diode behaves like a floating battery. The voltage changes at the collector are reproduced at the output with negligible attenuation. This second circuit requires less input-to-collector amplification.

The larger collector voltage swing for circuit (*a*) dictates the use of the less desirable clamping arrangement shown. The speed of the two circuits is comparable if the minimum current through the breakdown diode is at least 0.5 milliampere or thereabouts. The "not" circuit also performs the function of an amplifier. The circuit is designed to operate with input signals which are at least as "good" as the defining levels, v_{zero} and v_{one}, but it produces output signals which meet the requirements for the output from a TSD. It thus restores the margin which may have been lost in the logic circuitry.

Tests for dc voltage levels and transient response characteristics were made with the "not" circuits and the diode logic circuits. These tests were intended primarily to verify the design calculations and to disclose any factor which had been overlooked in the theoretical study.

Much more extensive tests were made with the TSD, since it was felt that this was the most important building block circuit, and that its design was less straightforward than that of the logic circuits. The tolerance requirements which were placed upon the transistor characteristics and other components for the TSD are listed below:

[6] G. L. Pearson and B. Sawyer, "Silicon *p-n* junction alloy diodes," Proc. IRE, vol. 40, pp. 1348–1351; November, 1952.

Requirements for TSD transistors:

$Q1$ and $Q2$ are point contact units:

$$1.6 < a < 3.0$$
$$15K < r_c < 30K$$
$$r_b < 0.8K.$$

$Q3$ and $Q4$ are junction units, either *p-n-p* or *n-p-n*:

$$a \geqq 0.95$$
$$r_b < 1K.$$

$Q5$ and $Q6$ are *n-p-n* junction units:

$$0.9 \leqq a \leqq 0.955$$
$$r_b < 0.8K.$$

The time required to change the state of the TSD varies from about 1 to 3 microseconds. The logic circuits respond to waveforms of this speed without introducing any appreciable additional delay.

The first step was to verify that the flip-flop circuit would be bistable with a wide variety of transistors. It would have been desirable to make extensive tests with a large number of transistors which met the specifications for TSD transistors. Unfortunately, the limited number of transistors on hand made this impossible. The first test was made with a group of 10 point-contact units which had seen considerable previous use. Only about half of the possible combinations of two transistors operated in a bistable manner in the flip-flop circuit. Measurements of these 10 units indicated that their characteristics had altered considerably during previous use, and only a few transistors still met specifications. The bistability test was repeated with two groups of 10 transistors each, which had not been previously used. Only a very few of the possible pairs in each group were not bistable. Subsequent measurements showed that at least one unit of each of these latter pairs had extremely poor characteristics in one respect or another (e.g., r_c values of 7,000 ohms, alpha values of 1.2).

A three-bit shift register was next constructed to test the dynamic characteristics of the circuits. Its logical diagram is shown in Fig. 7. The two sets of TSD's are necessary for an asynchronous shift register. Because there were not sufficient transistors available to build a truly asynchronous shift register, pulse trains from a separate circuit were used for synchronous control of the shifting operations. These pulses are indicated on the figure.

Since the left end of the register is connected to the right end, a three-bit number can be placed in the register and it will then cycle continuously through the register as successive right-shift operations are performed. An oscilloscope can be used to monitor the output of one of the TSD's, and a failure will result in a change in the pattern on the oscilloscope.

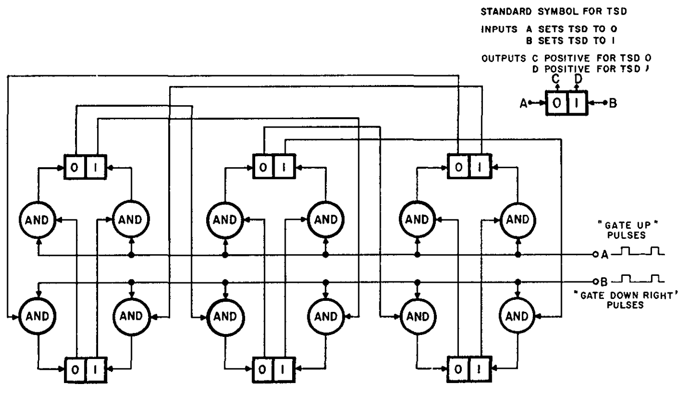


Fig. 7—Logical diagram for a three-bit shift register.

Error-free runs of more than two hours were obtained with this circuit, with a period of about 40 microseconds for a complete right-shift operation. This is of the order of 10^8 operations, which is a disappointing figure for a circuit of this simplicity. One factor was felt to be primarily responsible. The circuit was unusually sensitive to slight changes in transistors, since several of the point-contact transistors being used failed to meet the specifications in one particular or another. Initially the circuits would operate properly, since all of the other parameters were well within the limiting values, and there was sufficient margin to compensate for the substandard units. As soon as temperature variations or other causes made significant changes in the transistor characteristics, the additional margin would be used up, and the substandard transistors would cause a failure.

It was difficult to correct this condition for two reasons. First, it was difficult to locate the source of the failure, since the register was not operating asynchronously. In addition, there were not enough transistors on hand to replace all of the marginal units.

Two conclusions were drawn from the results of these tests. First, that the technique described will lead to successful circuits if two conditions are satisfied:

1. The point-contact transistors are sufficiently ideal to be represented by the three-region *N*-Curve.
2. The transistor parameters are actually within the range given by the manufacturer's data sheet.

The second conclusion was that a large percentage of the transistors available[7] did not satisfy these criteria. A three-region approximation was not adequate for many units; a more elaborate design technique would be required for these transistors. The manufacturer's specification sheet often gave only an approximate idea of the *initial* characteristics of the transistors. The type of circuits described here are not practical unless the transistors do meet the above criteria.

Acknowledgments

This work was carried out at the University of

[7] These tests were made in September 1954, with units which had been obtained between November 1953 and August 1954.

Illinois. It was partially supported by funds from ONR, on contract N6ori-71, Project ONR NR 048094, and from the Radio Corporation of America, through an RCA fellowship. Professors R. E. Meagher and G. H. Fett offered considerable guidance and assistance. H. N. Yu of the Digital Computer Laboratory of the University did much of the experimental work. Portions of this material were submitted in partial fulfillment of the Ph.D. degree in the Department of Electrical Engineering.

A Theorem on SPDT Switching Circuits

B. D. RUDIN†

Summary—**It is first demonstrated that networks of SPDT (Single Pole Double Throw) switches can be represented abstractly with a Boolean algebra in much the same way as networks of SPST (Single Pole Single Throw) switches have been treated by Shannon, provided a certain set of network wiring rules is established. A theorem is then proved which shows that all networks represented in this way can be handled as though each network were itself a single SPDT switch. Proof of a corollary then allows discussion of techniques of network construction. The component minimization problem is also discussed and several detailed examples are given.**

Introduction

THE ANALYSIS and synthesis of two-terminal switching circuits with the aid of a Boolean algebra has been treated extensively by C. E. Shannon and others[1–3] and is now a commonplace procedure which has been available for a number of years. Brevity demands that it be assumed here that the reader is reasonably familiar with two-terminal circuit techniques as given in the references.

For purposes of this paper, only certain general features of the analytical methods for two-terminal networks need be noted, and they are as follows:

1. The basic network element is an SPST (Single Pole Single Throw) switch.
2. There are just two ways to connect two elements together—in series or in parallel.
3. Any two-terminal switching network composed of SPST elements is itself an SPST switch.

The third feature assures us that complicated networks may be built up without a mass of detail since large segments of the network can be reduced analytically and diagrammatically to single primitive elements. This is not surprising, for in any Boolean algebra a Boolean function or polynomial of elements of the algebra is also an element of the algebra. The second feature is a statement of network wiring rules. Series and parallel can be made to correspond to product and sum in the descriptive algebra; 0 and 1 can denote open and closed circuits, respectively. (This is the opposite of Shannon's notation, but is of course just as good.)

In the physical realization of a two-terminal switching or relay network, SPDT (Single Pole Double Throw) elements are introduced wherever their use is indicated for reasons of economy. With relays, for example, the addition of a contact point is cheaper than the addition of a spring and contact point. As long as the network is relatively simple, the places for introduction of SPDT elements are easily found: they exist wherever an element and its negation are wired in parallel. Fig. 1 gives some simple SPST networks, showing the conventions and also showing the insertion of an SPDT element.

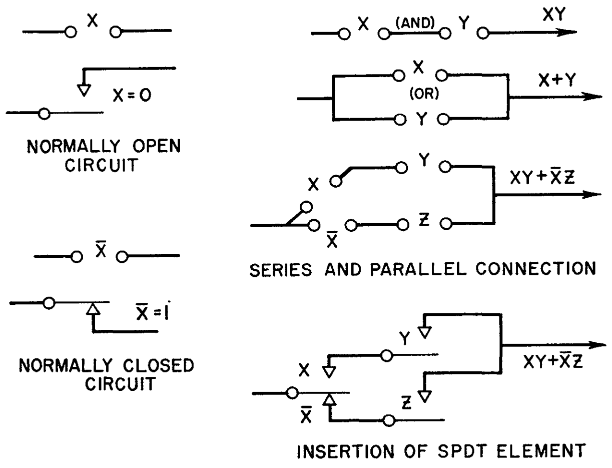


Fig. 1

Now, in the design of multifunctional networks one often finds that the negation of entire networks is called for. Using two-terminal methods, the designer develops the network and its dual separately and then proceeds to realize them as best he can. This can be quite a task. Fig. 2 shows an example of a selective two-terminal net-

† Lockheed Aircraft Corp., Van Nuys, Calif.

[1] C. E. Shannon, "A symbolic analysis of relay and switching circuits," *Proc. AIEE*, vol. 57, pp. 713–723; 1938.

[2] C. E. Shannon, "The synthesis of two-terminal switching circuits," *Bell Sys. Tech. Jour.*, vol. 28, pp. 59–98; January, 1949.

[3] W. Keister, A. E. Ritchie, S. H. Washburn, "The Design of Switching Circuits," D. Van Nostrand Co., Inc., New York, N. Y., pp. 69–103; 1951.

work and its dual taken from Shannon's first paper.[4] The desired network (top figure) requires 14 SPST elements, and even if SPDT elements be inserted in indicated places, it is hard to see further simplifications. However, a formal analytical procedure exists by which both of these networks can be reduced to a single three-terminal network containing nine SPDT elements (Fig. 3). This example will be treated at length later in this paper, after such an analytical procedure for three-terminal SPDT networks has been devised.

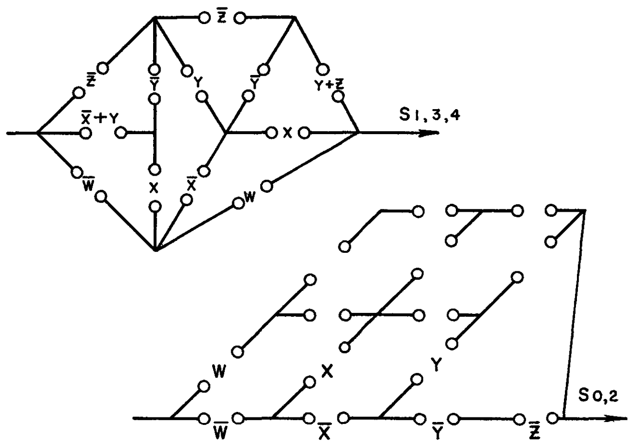


Fig. 2

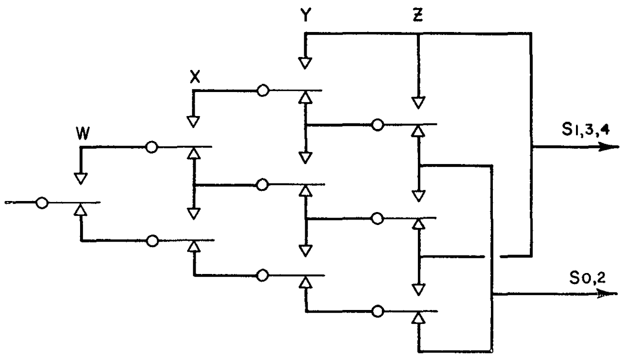


Fig. 3

SPDT Switching Networks

The objective to be sought in what follows is the provision of an analytical technique for networks of SPDT switches having features analogous to those given above for SPST networks. The desired features are:

1. The basic network element is a SPDT switch.
2. There is an unambiguous set of rules for connecting two elements.
3. Any three-terminal switching network composed of SPDT elements is itself an SPDT switch.

The symbolism to be used for an SPDT switch has already been shown in Fig. 1. Zero and 1 again mean open and closed circuits, respectively. The normally closed contact will always be designated with the bar for negation.

A Boolean algebra can be set up as an analytical tool for the design of SPDT networks as well as for SPST networks, since both SPST and SPDT switches have the necessary two-valued property. A rigorous postulational treatment need not be given here; rather, an emphasis will be placed on the correspondence between logical equations and circuits. The first step will naturally be the setting up of a circuit correspondence for the Boolean product and sum. This was not difficult with SPST elements; however, with SPDT elements ambiguities are immediately encountered. As an example, consider the problem of wiring a two-terminal network for the Boolean sum $X+Y$ using two SPDT switches. Figs. 4a and 4b represent the two ways of constructing the circuit. In Figs. 4c and 4d, the remaining terminals in the circuits of 4a and 4b have been used to form three-terminal circuits with the algebraic descriptions of the third terminals included. The advantages of circuit 4d are immediately apparent, for this circuit has the desired third feature specified above: the network itself is an SPDT switch by virtue of the algebraic identity

$$\overline{X+Y} = \bar{X}\bar{Y}.$$

From this example and others which are easily obtained, the following rule is suggested:

Wiring Rule I—Poles must not be wired in parallel.

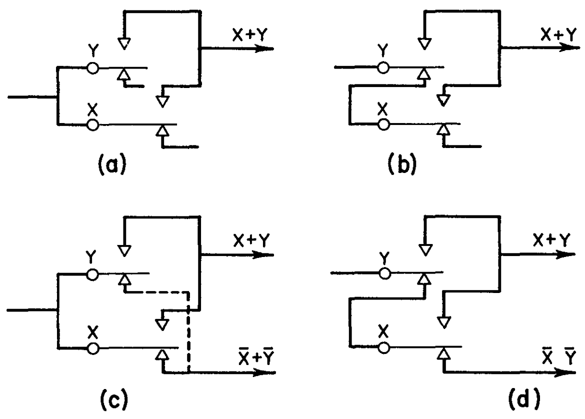


Fig. 4

As another example of an ambiguous wiring situation, consider the problem of obtaining a two-terminal network using SPDT elements for the algebraic expression of the logical notion "exclusive and,"

$$A \dot{\top} B = AB + \bar{A}\bar{B}.$$

This circuit can be obtained using as few as two and as many as four elements, and some of the possible solutions are shown in Figs. 5a, b, c, d. Note, however, that the four-element solution in 5d is immediately ruled out by Wiring Rule I. We would also like to rule out the circuit in 5a, since it is a pure two-terminal solution, it being impossible to obtain a third terminal at which the network negation might be obtained for realization of

[4] Shannon, "A symbolic analysis of relay and switching circuits," p. 722.

our third desired feature. The negation of "exclusive and" is "exclusive or"; symbolically,

$$\overline{A \underset{\cdot}{\top} B} = A \underset{\cdot}{\perp} B = A\bar{B} + \bar{A}B.$$

Both the networks 5b and 5c satisfy the SPDT requirement. This ambiguity can be removed with

Wiring Rule II—In a given network of SPDT elements all but one of the poles must be wired in series with non-poles.

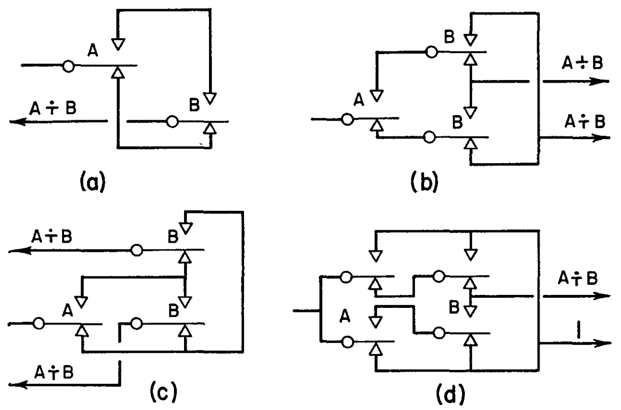


Fig. 5

This eliminates both the solutions 5a and 5c in the example, leaving only 5b. Two A elements and one B element may be used instead of the arrangement shown, but this would only be done if it were important to conserve B elements. The one open pole mentioned in the rule will always be the pole terminal of a three-terminal network. We are now in a position to state and prove the following:

Theorem—Given a two-terminal switching network constructed of SPDT elements in accordance with Wiring Rules I and II above, which represents a Boolean function F of n variables, the negation of F can be wired through the same elements which represent F without affecting the network for F.

Proof: There are just ten distinct Boolean functions of two variables which will produce just five distinct three-terminal networks giving the ten functions in negation pairs. The functions and networks are shown in Figs. 4d, 5b, and 6. Hence, the theorem is true for $n=2$. Now, assume the theorem is true for functions of $n-1$ variables and consider a function F of n variables, $X_1, X_2, \cdots, X_n$. By the expansion theorem of Boolean algebra,

$$F(X_1, \cdots, X_n) = X_nF(X_1, \cdots, X_{n-1}, 1) + \bar{X}_nF(X_1, \cdots, X_{n-1}, 0).$$

Let

$$f_1 = F(X_1, \cdots, X_{n-1}, 1),$$
$$f_2 = F(X_1, \cdots, X_{n-1}, 0).$$

Since the theorem holds for $n-1$ variables, the function f_1 and its negation $\bar{f}_1$ can be represented by a single SPDT element. The same is true for f_2. Fig. 7 shows the realization of the network for F and $\bar{F}$ through the same elements. The expansion could have been made about any of the n variables and the result arrived at would be the same. Furthermore, the Wiring Rules guarantee that the original network for F must have been achieved by expansion around one of the variables. Consequently, the theorem has been proved by finite induction.

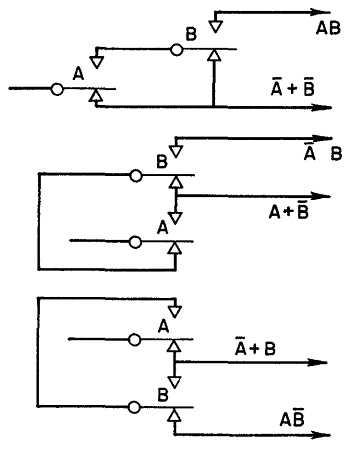


Fig. 6

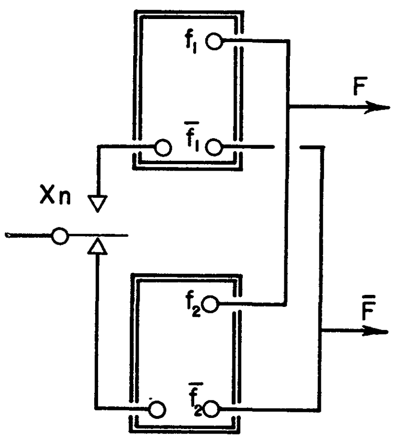


Fig. 7

Corollary—Given the conditions of the theorem, the negation network $\bar{F}$ may be obtained by connecting in parallel all the unused contact points on the switches used to construct the network for F.

The proof of this statement is almost identical to that of the theorem so that time need not be spent on it. The corollary provides us with a concise method of instrumenting the stated objectives for SPDT networks which have now been achieved.

Design Considerations

As an illustration of the application of the theorem and corollary, we may use the example of Figs. 2 and 3. The original problem was to obtain a network which provided a closed circuit whenever any one, three, or all four of relays w, x, y, and z were picked up. The algebraic expression for this network is a so-called symmetric function of the variables which is usually written

$$S_{1,3,4}(w, x, y, z) = wxyz + \bar{w}xyz + w\bar{x}yz + wx\bar{y}z + wxy\bar{z} + w\bar{x}\bar{y}\bar{z} + \bar{w}x\bar{y}\bar{z} + \bar{w}\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}z.$$

The networks for realization of symmetric functions have been discussed at length by Shannon and others.[5,6] These networks are usually called "trees." It is easily recognized that the network for

$$S_{0,2}(w, x, y, z) = \bar{w}\bar{x}\bar{y}\bar{z} + \bar{w}\bar{x}yz + \bar{w}x\bar{y}z + \bar{w}xy\bar{z} + w\bar{x}\bar{y}z + w\bar{x}y\bar{z} + wx\bar{y}\bar{z},$$

which provides a closed circuit whenever none or any two of the relays are picked up, gives a tree which contains fewer SPST elements than the desired tree would. This network is at the bottom of Fig. 2. It is also readily seen that the expressions given above are duals of each other. Since "tree" networks are planar SPST networks, the dual SPST network can be derived in the usual way, and this was done to obtain the desired network at the top of Fig. 2 from the one at the bottom.

Trees are naturally constructed with SPDT elements. If this is done in the example, the corollary tells us that the desired network can be obtained merely by paralleling the unused contact points in the dual. This is what was done in the network of Fig. 3. Furthermore, the theorem tells us that the labor spent in obtaining the dual network first was wasted. The tree for $S_{1,3,4}(w, x, y, z)$ gives the same SPDT network, for a function and its dual must be obtainable in the same network.

For the design of multifunctional networks the following set of rules seem to be suggested by the above results:[7]

1. Write the algebraic expression for the system, being careful to introduce new dependent variables for each time sequence step.
2. Simplify the expressions if possible.
3. *Search for duals.* Often an entire expression for one of the independent variables may be removable by writing it as the dual of a combination of other dependent variables.
4. Draw the circuits for all dual portions, combining networks where possible.
5. Fit the remainder of the network in wherever convenient. Note that portions of the final network may contain SPST elements.

These rules are essentially not different from those given by Shannon. The only addition is the third rule, which may save time in arriving at a suitably economical network to meet a given situation. The most important thing to remember is that it is just as easy to draw an SPDT network as it is an SPST network, and the labor of obtaining the dual is eliminated. However, the theorem does *not* guarantee that a three-terminal network developed by use of the wiring rules and the corollary is the most economical network from the component minimization point of view. It is true that minimal networks have been found for certain algebraic forms (i.e., symmetric functions), but ingenuity will generally be required before a final network is completed. This is certainly true in the design of multifunctional networks which could contain both SPST and SPDT elements.

As an example, consider the problem of obtaining the network for

$$U = \bar{X}A(\bar{B} + \bar{Y}) + XCY$$
$$V = \bar{X}BY + XCY$$
$$W = \bar{X}\bar{A}(\bar{B} + \bar{Y}) + X(\bar{C} + \bar{Y})$$
$$\bar{W} = \bar{X}(A + BY) + XCY.$$

This problem was encountered during the construction of a computing system. The network which was used is shown in Fig. 8. The result was obtained by first drawing the three-terminal network for W and $\bar{W}$, and it was then observed that U and V could be conveniently obtained as offshoots of the three-terminal network. One-way elements were used to avoid back circuits. It is possible to wire a combined SPST and SPDT network for this problem without the use of one-way elements, but in this case the given network was felt to be the best solution. Many arrangements were tried before this network was accepted.

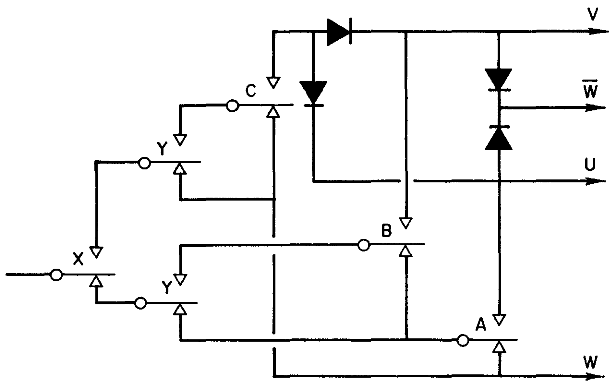


Fig. 8

It is hoped that the discussion given above has not been too brief. The basic objective has been to give the logical network designer some additional tools to help save designing time and possibly to help in the simplification of networks by component elimination. If the details given above are lost through hasty presentation, it is hoped that at least the broad idea of the application of the theorem has been put across. If this is so, the facile designer should have no trouble in developing detail enough to solve his particular problems. The author has found the techniques described above to be very helpful and hopes that others may benefit as well.

[5] Shannon, "A symbolic analysis of relay and switching circuits," p. 719.
[6] Keister, Ritchie, Washburn, *op. cit.*, ch. 4.
[7] Shannon, "A symbolic analysis of relay and switching circuits," p. 721.