IX. PROCESSING AND TRANSMISSION OF INFORMATION*

Prof. D. N. Arden
Prof. E. Arthurs
Prof. J. B. Dennis
Prof. M. Eden
Prof. P. Elias
Prof. R. M. Fano
Prof. R. G. Gallager
Prof. E. M. Hofstetter
Prof. D. A. Huffman
Prof. W. F. Schreiber
Prof. C. E. Shannon
Prof. J. M. Wozencraft
S. Asano
G. Cheadle
J. E. Cunningham
P. M. Ebert
H. A. Ernst
E. F. Ferretti
T. J. Goblick, Jr.
U. F. Gronemann
F. C. Hennie III
T. S. Huang
F. Jelinek
T. Kailath
L. Kleinrock
H. H. Loomis, Jr.
J. L. Massey
J. W. Pan
D. L. Reich
L. G. Roberts
O. J. Tretiak
J. Ziv

A. PICTURE-PROCESSING RESEARCH

1. COMPUTER PROCESSING CHAIN

To facilitate the study of images and image-coding methods, we are assembling a research device that will enable us to make digital computer tapes from original photographs and vice versa. This was described in Quarterly Progress Report No. 61 (pages 133-135).

We had intended to use for this device the same flying-spot scanner optical system

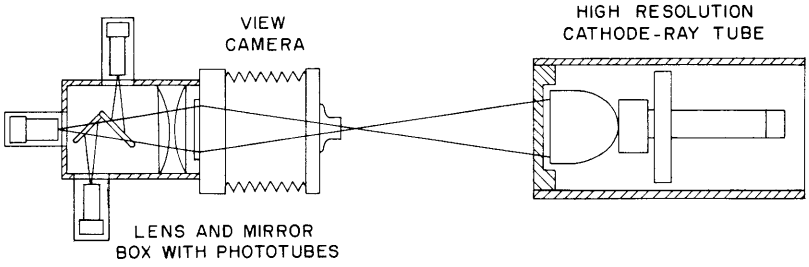


Fig. IX-1. Digital scanner optical system.

used in the real-time color system project described below, but this has turned out to be operationally inconvenient. Therefore, during the past quarter, we have designed and constructed a new scanner optical system expressly for the digital system. It is built around a commercial view camera. As shown in Fig. IX-1, the camera back has been replaced by a special assembly that holds the picture, condenser lenses, mirrors, and phototubes, of which three are provided. In this way, it will be possible to obtain, from a single color slide, three simultaneous video signals representing the three color components of the

*This research was supported in part by Purchase Order DDL B-00306 with Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology, with the joint support of the U. S. Army, Navy, and Air Force under Air Force Contract AF19(604)-5200.

image. For each point in the picture the three video signals will be sampled, quantized, and tape-recorded sequentially. Subsequent processing operations on the color image can be carried out in the computer in much the same way as for monochromatic images.

When the processed tape is played back, the image will be displayed on the same scanner tube. The camera back will then be replaced by a 4 × 5 Polaroid film holder for recording the image. Color film will be used for color pictures. Successive exposures of the same film through three color filters will then permit the recording of a full color image.

The new scanner has been assembled and is now undergoing tests.

J. E. Cunningham, U. F. Gronemann, T. S. Huang,
J. W. Pan, O. J. Tretiak, W. F. Schreiber

2. COLOR DISPLAY BY THE LAND PROCESS

A great deal of interest has been shown in the papers by Edwin H. Land, and by others, concerning his work in the field of two-color systems. The scientific community is sharply divided with respect to the ultimate possibility of satisfactory color rendition by such a process. We have undertaken an experiment in this field, primarily because of the added insight it may give us into the mechanism of color perception when observing images, as distinguished from the normal color-matching situation, which is the basis of tristimulus colorimetry.

We have constructed a real-time closed-circuit color television system, using a simultaneous pick-up flying-spot scanner and a field-sequential display. The display operates at 144 fields/sec; this gives a color frame rate of 24 per second with 525 lines per frame. In three-color operation, the display sequence is red, blue, green; and in the two-color mode, red, red, white.

Color pictures in both modes have recently been observed with this apparatus. Both types of picture had certain defects resulting from idiosyncrasies of the instrumentation; these will be remedied. However, certain conclusions about the two-color picture can be drawn. The color rendition was substantially inferior to that of the three-color system, and was very similar to the demonstrations that we have seen by the "anti-Land" scientists. The pictures were very inferior to those produced by Land himself. This leads to the conclusion that the process which we (and previous demonstrators) are using is not precisely comparable to that used by Land. We believe – and this is, at the present time rather speculative – that the tone scale of the separate components may be of crucial importance to the rendition of the noncolorimetric hues, such as yellow and blue. In the next few months we hope to investigate these matters more carefully.

S. Asano, W. F. Schreiber

B. DETERMINATION OF R_{comp} FOR THE GENERAL MEMORYLESS ASYMMETRIC CHANNEL

Reiffen (1) has defined a computation cutoff rate, R_{comp}, for a discrete memoryless channel. However, only an upper bound on R_{comp} was presented.

In this report we present a lower bound on R_{comp} for the general memoryless (continuous or discrete) channel. This report is not intended to be self-contained. The reader is assumed to be familiar with Reiffen's report.

Average Number of Computations

For a particular decoding, the number of computations required to process the incorrect subset is

$$N = \Sigma_w N(w) \tag{1}$$

where N(w) is the number of computations required to extend the correct subset from w to w + 1.

Using bars to denote averages, we have

$$\overline{N} = \Sigma_w \overline{N(w)} \tag{2}$$

But

$$\overline{N(w)} \leq \sum_{\lambda, j} e^{wR} P_r\left[Y_w \geq D_w^{(j)}; X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right] \tag{3}$$

where $\left\{Y_w \geq D_w^{(j)}\right\}$ is the event in which a member of the incorrect set is bigger than the cutoff level $D_w^{(j)}$ of the j^{th} criterion, at length w, and therefore will not be discarded.

Also, $\left\{X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right\}$ is the event in which the correct message was indeed discarded at some length $\lambda w \leq n$ when the $(j-1)^{th}$ criterion was used. In other words, this is the event in which the j^{th} criterion is to be used in this step.

$D_w^{(j)}$ is given by

$$P\left(X_w \leq D_w^j\right) \leq e^{-k_j} \tag{4}$$

and we would like to upper-bound

$$P_r\left[Y_w \geq D_w^{(j)}; X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right]$$

by an exponential term so that

$$P_r\left[Y_w \geq D_w^{(j)}; X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right] \leq Be^{-R^* w} \tag{5}$$

where B is a constant that is independent of w and λ, so that

$$\overline{N} \leq \sum_{\ell=1}^{w} \sum_{j,\lambda} Ke^{(R-R^*)w} \tag{6}$$

The minimum value of R^*, over all w, λ, and j is called "R_{comp}." Thus,

$$R^* \geq R_{comp} \tag{7}$$

Now, $X_w = \sum_{\ell=1}^{w} x_\ell$ is the sum of w independent random variables, of the form:

$$x = \log \frac{q_i(j)}{Q_j} \qquad Q_j = \Sigma_i p_i q_i(j) \tag{8}$$

where p_i is the probability of the transmitted letter i, and $q_i(j)$ is the probability j that was received, given that i was transmitted. Thus, by the use of Chernoff Bound,

$$P\left(X_w \leq D_w^{(j)}\right) \leq e^{w(\mu(s)-s\mu'(s))} \quad s \leq 0 \tag{9}$$

where

$$\mu(s) = \ln \sum_{i,j} p_i q_i(j)\, e^{sx} = \sum_{i,j} p_i q_i(j)^{1+s} Q_j^{-s}$$

and

$$\mu'(s) = \frac{D_w^{(j)}}{w}$$

Thus by inequalities 4 and 9,

$$p\left(X_{\lambda w} \leq D_{\lambda w}^{(j)}\right) \leq e^{\lambda w(\mu(s)-s\mu'(s))} = e^{-k_j} \tag{10a}$$

and

$$D_{\lambda w}^{j} = \lambda w \mu'(s) \tag{10b}$$

where s is determined as the solution of the equation

$$\frac{k_j}{\lambda w} = s\mu'(s) - \mu(s) \tag{10c}$$

In the same way,

$$Y_w = \sum_{\ell=1}^{w} y_\ell$$

is the sum of w independent random variables of the form:

$$y = \log \frac{q_k(j)}{Q_j} \tag{11}$$

where k is some input symbol, and

$$p\left(Y_w \geq D_w^j\right) \leq e^{w(\gamma(t)-t\gamma'(t))} \quad t \geq 0 \tag{12}$$

with

$$\gamma(t) = \ln \sum_{j,i,k} p_k p_i q_i(j)\, e^{ty} = \sum_{k,j} p_k Q_j e^{ty} = \sum_{k,j} p_k Q_j^{1-t} q_k^t(j)$$

and $D_w^{(j)} = w\mu'(t)$.

Now, returning to inequality 5,

$$\Pr\left[Y_w \geq D_w^{(j)}; X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right] \leq \Pr\left[Y_w \geq D_w^{(j)}\right] \tag{13a}$$

Also,

$$\Pr\left[Y_w \geq D_w^{(j)}; X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right] \leq \Pr\left[X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right] \tag{13b}$$

Thus, by inequalities 13, 10, and 12,

$$\begin{aligned}\Pr\left[Y_w \geq D_w^{(j)}; X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right] &\leq \min\left\{\Pr\left[Y_w \geq D_w^{(j)}\right];\ P_r\left[X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right]\right\} \\ &\leq \min\left\{e^{\gamma(t)-t\gamma'(t)};\ e^{-k_{j-1}}\right\}\end{aligned} \tag{14}$$

Let $k_j = k_{j-1} - \Delta$; $\Delta \geq 0$. Thus

$$e^{-k_{j-1}} = e^{\Delta}\, e^{-k_j} = e^{\Delta}\, e^{(\mu(s)-s\mu'(s))w}$$

where

$$\left.\begin{aligned}\mu'(s) &= \frac{D_w^{(j)}}{w} \\ \text{and} \\ \mu(s) - s\mu'(s) &= -\frac{K^{(j)}}{w}\end{aligned}\right\} \tag{15}$$

Therefore, by inequalities 5 and 14 and Eq. 15,

$$P\left(Y_w \geq D_w^{(j)}; X_{\lambda w} \leq D_{\lambda w}^{(j-1)}\right) \leq \min\left\{e^{w[\gamma(t)-t\gamma'(t)]}; e^{\Delta} e^{w[\mu(s)-s\mu'(s)]}; e^{\Delta} e^{w[\mu(s)-s\mu'(s)]}\right\}$$

$$\leq e^{\Delta} \min\left\{e^{w[\gamma(t)-t\gamma'(t)]}; e^{w[\mu(s)-s\mu'(s)]}\right\}$$

$$= e^{\Delta} e^{-wR^*}$$

Thus,

$$R^* = \max\left\{-\gamma(t)+t\gamma'(t); -\mu(s)+s\mu'(s)\right\} \tag{16a}$$

where

$$\mu(s) = \ln \sum_{ji} p_i q_i(j) e^{sx} = \sum_{ij} p_i q_i(j)^{1+s} Q_j^{-s} \tag{16b}$$

$$\gamma(t) = \ln \sum_{jk} p_k Q_j e^{ty} = \sum_{kj} p_k Q_j^{1-t} q_k(j)^t \tag{16c}$$

and

$$\mu'(s) = \gamma'(t) = \frac{D_w^{(j)}}{w} \tag{16d}$$

Thus

$$R^* \geq \frac{1}{2}\left\{-\gamma(t)+t\gamma'(t)-\mu(s)+s\mu'(s)\right\}$$

$$R_{comp} = \min\left\{R^*\right\} \geq \min\left\{\frac{1}{2}\left[t\mu'(t)-\mu(t)+s\mu'(s)-\mu(s)\right]\right\} \tag{17}$$

Now, by Eqs. 16,

$$\mu'(s) = \frac{\sum_{ij} p_i q_i(j) \left(\frac{q_i(j)}{Q_j}\right)^s \log\left(\frac{q_i(j)}{Q_j}\right)}{\sum_{i,j} p_i q_i(j) \left(\frac{q_i(j)}{Q_j}\right)^s}$$

$$\gamma'(t) = \frac{\sum_{kj} p_i Q_j \left(\frac{q_i(j)}{\partial j}\right)^t \log\left(\frac{q_i(j)}{\partial j}\right)}{\sum_{k,j} p_i Q_j \left(\frac{q_i(j)}{Q_j}\right)^t}$$

If we let $t = 1 + s$, we have the result: $\gamma'(t) = \mu'(s)$ (also $\gamma(t) = \mu(s)$). Hence

$$R_{comp} \geq \min \left\{ \frac{1}{2} [(1+2s)\mu'(s) - 2\mu(s)] \right\} \tag{18}$$

The minimum occurs at that s, for which

$$[(1+2s)\mu'(s) - 2\mu(s)]' = 0$$

$$(1+2s)\mu''(s) + 2\mu'(s) - 2\mu'(s) = 0$$

$$(1+2s)\mu''(s) = 0$$

$$s = -\frac{1}{2}$$

Also, $[(1+2s)\mu'(s) - 2\mu(s)]'' = 2\mu''\left(-\frac{1}{2}\right) \geq 0$, since $\mu''\left(-\frac{1}{2}\right)$ is the variance (see Fano (2)) of a random variable. Thus, $s = -\frac{1}{2}$ is indeed a minimum point. Thus

$$R_{comp} \geq -\mu\left(-\frac{1}{2}\right)$$

Now,

$$\mu\left(-\frac{1}{2}\right) = \ln \sum_{i,j} p_i \sqrt{Q_j} [q_i(j)]^{1/2}$$

and therefore

$$2\mu\left(-\frac{1}{2}\right) = \ln \left\{ \sum_{i,j} p_i \sqrt{Q_j} [q_i(j)]^{1/2} \right\}^2$$

where

$$f_j = \sum_i p_i (q_i(j))^{1/2}$$

By the Schwarz inequality,

$$\left\{ \sum_j f_j \sqrt{Q_j} \right\}^2 \leq \sum_j f_j^2 \sum_j \left(\sqrt{Q_j} \right)^2$$

Thus

$$2\mu\left(-\frac{1}{2}\right) \leq \ln \left[\sum_j f_j^2 \sum_j Q_j \right]$$

but

$$\sum_j Q_j = \sum_{j,i} p_i q_i(j) = 1$$

so that

$$2\mu\left(-\frac{1}{2}\right) \leq \ln \sum_j f_j^2 = \ln \sum_j \left[\sum_i p_i q_i(j)^{1/2} \right]^2$$

and

$$R_{comp} \geq -\frac{1}{2} \ln \sum_j \left[\sum_i p_i q_i(j)^{1/2}\right]^2$$

But $-\ln \sum_j \left[\sum_i p_i q_i(j)^{1/2}\right]^2 = E(0)$, where $E(0)$ is the zero-rate exponent of the upper bound on the average probability of error p_e, for the same channel, when an optimal decoding scheme is used (2).

$$p_e \leq 2e^{-nE(R)}$$

Thus

$$R_{comp} \geq \frac{1}{2} E(0) \qquad \text{Q. E. D.}$$

This research was supported in part by a fellowship from the Government of Israel.

J. Ziv

References

1. B. Reiffen, Determination of R Cutoff for the Asymmetric Channel, Report 25G-0021 Rev. 1, Lincoln Laboratory, M. I. T.,

2. R. M. Fano, Transmission of Information (The M. I. T. Press, Cambridge, Mass., and John Wiley and Sons, Inc., New York, 1961); see Chapter 9.

C. INFORMATION FLOW IN LARGE COMMUNICATION NETS

The purpose of this investigation is to consider the problems associated with information flow in large communication nets. The nets considered consist of nodes that receive, sort, store, and transmit messages entering and leaving by way of the links (one-way channels that connect the nodes together).

Very little effort has been devoted to these problems in published works, although there is a clear practical need for an understanding of these nets. Jackson (1) has considered a class of related problems dealing with a system of departments in which messages travel between the departments according to a probability measure assigned to each link (including sources and sinks). His results show that it is possible to break the system down into independent elementary departments.

In the present investigation, a number of results have been obtained for systems that are similar to Jackson's, but altered enough to represent a communication net. In particular, it can be shown that it is not possible, in general, to assign an arbitrary probability measure on the use of the channels (as Jackson assumed for his departments), although a fair approximation to this situation can be developed. Moreover, there has been added, as a constraint on the communication net, the necessity for a

message to leave the system upon delivery at its final destination (a constraint not included by Jackson). With this additional constraint, it has been shown that for one class of systems investigated, the solution takes on a form that is almost identical to Jackson's, and insures the independence of the nodes.

It has also been shown that for a general class of systems, all traffic leaving nodes in the net is Poisson in nature (that is, the inter-departure times of messages are independent, and obey an exponential distribution).

Another general result involves the definition of p, the utilization factor, of a one-input, N-output node, where the input traffic is Poisson, with mean rate λ, the message lengths are exponentially distributed with mean length $1/\mu$, and the routing and queueing discipline is completely arbitrary. The capacity assigned to each output channel is arbitrary, subject to the condition that the total capacity on all channels must sum to C.

Let

P_n = Pr [n messages in a node in the steady state]

$\overline{C_n}$ = Expected value of the unused capacity, given n messages in the node,

$p = \lambda/\mu C$ = utilization factor (a common definition)

Then, it can be shown that

$$p = 1 - \sum_{n=0}^{\infty} (\overline{C_n}/C)P_n$$

provided that the steady state is achieved.

L. Kleinrock

References

1. J. R. Jackson, Networks of waiting lines, Operations Res. 5, 518 (1957).

D. DELAYED-LOGIC AND FINITE-STATE MACHINES

1. Introduction

The investigation described in this report was motivated by the contention that fast computers must be small. An implication of the results obtained is that this is not necessary if the only effect of large size is the increased propagation time of signals relative to the clock rate. There are other effects that are important – for example, reduction of bandwidth caused by dispersion along a long line – which are not discussed here. These are, generally, of less concern in present computer design, but a study of effects of this type would be interesting.

The methods used in the investigation are as intriguing to the writer as the results, and provide tools for the proof of general statements about automata.

2. Elementary Machines and Definite Events

We shall consider a machine to have several input terminals and a single output terminal. Each input terminal will accept, and the output terminal produce, a binary-valued signal, one value being denoted by zero and the other by one. Other conventions are possible, some of which can be reduced to this case; and some examples of such a reduction will be given. In particular, more than one output terminal can be considered, but if economy of circuit design is not an issue, extension to this case is straightforward.

The set of possible input configurations will sometimes be called the "input alphabet" and, for economy of notation, they may be denoted by lower case letters or decimal integers.

In general, machines are constructed from a finite number of copies of a finite number of machines from a set (finite or infinite) of distinguished machines called devices, by identifying output terminals of some devices with (one or more) input terminals of others. It will be assumed that no distinct output terminals are identified. Such an arrangement, which may be called a net or circuit, defines a machine in the sense mentioned above if exactly one output terminal is left unconnected. The ordered set of unconnected input terminals will be called direct inputs to the machine, and the set of all possible binary input configurations will be denoted by a set of symbols $x_1, x_2, \ldots, x_m$ called the input alphabet for the machines. The set of all symbols of the input alphabet will be denoted by I.

An ordered set of input symbols will be called an input sequence. In this report we shall consider only the synchronous case, characterized by the assumption that inputs are accepted at equally spaced instants of time called clock times, separated by an interval called the clock interval whose reciprocal is the input frequency. The clock interval can be taken to be one with full generality.

It will be assumed that a special signal is available at $t = 0$, called a starting pulse and denoted by λ. This pulse may be used to impose some initial conditions on the net at $t = 1$ by using input terminals not necessarily in the set of direct input terminals. All input sequences will be assumed to begin at $t = 1$.

If a device is replaced by a set of directed or oriented lines from its input terminals to its output terminal, a machine becomes a directed or oriented graph with the various terminals as nodes. A machine is said to be circle-free if it is not possible to start at an arbitrary node and return to the same node by following lines of the graph in the direction of their orientation.

The class of machines whose output at any clock time t is uniquely determined by the direct inputs at times $t - \ell + 1, t - \ell + 2, \ldots, t$ for some $\ell \geq 1$ will be called

elementary machines. To describe the potentialities and limitations of elementary machines it is convenient to introduce the notion of definite event, which is due to Kleene (1).

An event is a set of input sequences.

An event is definite if, for any input sequence and some non-negative integer ℓ, it can be determined whether or not it belongs to the set defining the definite event by inspection of, at most, its last ℓ symbols.

A finite set of input sequences (finite event) is a definite event, with ℓ the length of the longest sequence of the set

Since the input sequences constituting a definite event are characterized by the final ℓ symbols of the sequence, for input alphabets consisting of all or a subset of the combinations of n binary digits (n-input machines), definite events can be described by a Boolian function of the n inputs for each of ℓ consecutive times ending with time t, in the sense that a sequence ending at time t is a member of the event if and only if the value of the Boolian function is one. Such a function will be called the characteristic function of the event, or less formally, the Boolian description of the definite event.

For example, the characteristic function of the definite event consisting of all input sequences terminating in 00 or 1 for a machine with a single input A is $\bar{A}(t-1) \wedge \bar{A}(t) + A(t)$. If the event is the finite set consisting of only the sequences 00 and 1, the characteristic function is $\bar{A}(1) \wedge \bar{A}(2) + A(1)$.

From these definitions, it is apparent that a machine is elementary only if all input sequences resulting in an output of one at the time of the last input constitute a definite event. The machine is said to represent the definite event consisting of all such sequences.

It may be that the output at any time t is independent of the input symbols at $t, t-1, \ldots, t-d+1$, but is uniquely defined by the inputs at earlier times. Under these circumstances it is convenient, but not essential, to describe the action of the machine as representing, with delay d, the definite event consisting of all input sequences that cause an output of one d units of time after the last symbol in the sequence.

The set of devices described in Fig. IX-2 has played a predominant role in the theory of synthesis of machines. Their action can be described by labeling their terminals with distinct upper case letters, A, B, C, ..., possibly with subscripts, and denoting the signal at a given terminal, say A, at time t = j by the binary variable A(j).

This is a finite set of devices. In the interest of economy of notation, it is convenient to make use of a compact notation for certain combinations of these devices. For example, the devices of Fig. IX-3 are useful abbreviations.

All of these are elementary machines. The "and" gate represents (with zero delay) the set of all input sequences of pairs of binary digits terminating with a pair of ones. The "or" gate represents (with zero delay) all input sequences of pairs of binary digits

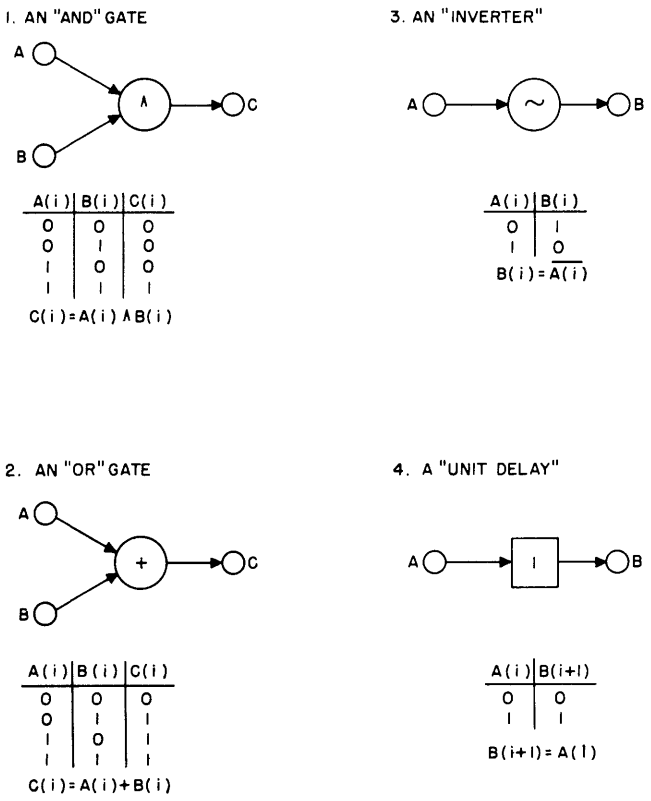


Fig. IX-2. Some common devices.

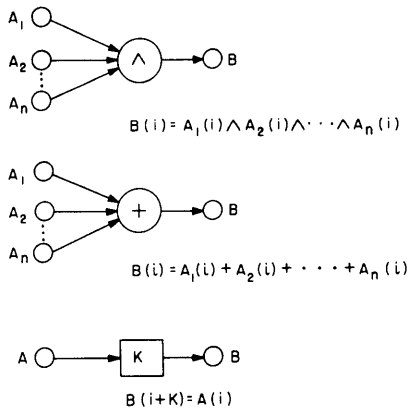


Fig. IX-3. Multiple input "and" gate, "or" gate, and K-unit delay.

terminating with either 01, 10 or 11. The unit delay represents with delay zero the set of all input sequences of binary digits whose next to last input is one or, equivalently, represents with delay one the set of all input sequences of binary digits terminating with a one. The "inverter" represents with delay zero all sequences terminating with zero. Hence all of these devices represent infinite definite events (called "noninitial" by Kleene).

The machine of Fig. IX-4 represents the finite set of binary sequences consisting of the single sequence 1, with delay zero. This is a finite event (called "initial" by Kleene) and hence definite.

The action of any elementary machine can be described by a table, or equivalently, a Boolian function, giving its outputs in terms of the inputs over a finite segment of the past. A set (finite or infinite) of elementary machines will be called complete if it is possible to construct a machine that represents any definite event with some delay from a finite number of copies of machines of the set.

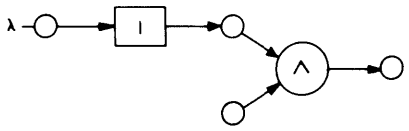


Fig. IX-4. A machine that represents a finite event.

The following theorem gives a rather obvious sufficient condition for completeness of a special class of elementary machines.

THEOREM 1. A set of elementary machines whose output at a given time depends only on the inputs at a single earlier time is complete if and only if

(a) the Boolian functions describing the output in terms of the inputs are complete for Boolian algebra;

(b) it is possible to synthesize either a unit delay or two relatively prime delays.

PROOF. Let the machines be M_j (finite or infinite in number) and their outputs at time t be expressed in terms of their inputs at time $t - d_j$ by the Boolian function $m_j[X_1(t-d_j), X_2(t-d_j), \ldots, X_{n_j}(t-d_j)]$, where, with full generality, the d_j have greatest common divisor one, and $X_1, X_2, \ldots, X_{n_j}$ denote the n_j distinct input lines to M_j. By (a) any Boolian function can be written in terms of the $m_j(x_1, x_2, \ldots, x_{n_j})$. Suppose that

$$m_j(Y_1(t-i_1),\ Y_2(t-i_2),\ \ldots,\ Y_{n_j}(t-i_{n_j}))$$

occurs in the Boolian description of some arbitrary definite event, where the $Y_1, Y_2, \ldots, Y_{n_j}$ denote the binary variables corresponding to (not necessarily distinct) positions in the binary code for the input alphabet. The function $m_j(Y_1(t-i_1), Y_2(t-i_2), \ldots, Y_{n_j}(t-i_{n_j}))$

can be computed by the machine M_j if $Y_K(t-i_K)$ can be delayed by $i_K(K=1, 2, \ldots, n_j)$, since in this event the signals will arrive at the inputs to M_j simultaneously at t. However, if the delays s_1 and s_2 are relatively prime or both equal to one, integers p and q can be found for which $ps_1 + qs_2 = 1$. It is an easy consequence of this result that any delay of the form $s_1s_2 + K$, $K \geq 0$, and, in particular, the delays $s_1s_2 + i_K$, can be generated. Hence the function in question can be computed with delay $s_1s_2 + d_j$, since all the i_K are ≥ 0.

In general, some argument, say the k^{th}, of a function m_j in the description of an arbitrary definite event may be the value of another function rather than a direct input. In the computation that has the M_j and delay elements, these values will be available only after considerable delays. This can be described by letting the corresponding i_K be negative. Under these circumstances, the magnitude m of the smallest i_j should be added to all of the arguments; this yields non-negative integers and permits the method described for the case of all arguments to be direct inputs to succeed, but increases the delay by the amount m. In general, this will give a representation for the arbitrary event with delay bounded by $s_1s_2 + \max t_i + d_j$ for each level of logical depth in the Boolian expression for the definite event in terms of the m_j, where the maximum is taken over all delays t_i with which functions m_i appearing as arguments of m_j can be realized.

As for the necessity of conditions (a) and (b), it is apparent that any Boolian function can be represented with some delay because it is a definite event. If all delays are allowed to approach zero, this representation which is condition (a), will still be valid.

Since the set of devices is complete, the functions

$U(t) = 1 \qquad t \geq K \qquad$ for some $K \geq 0$

$B(t+d') = A(t) \qquad$ for some $d' \geq 0$

$E(t+d) = D(t) \wedge E(t-1) \qquad$ for some $d \geq 0$

can be synthesized because they correspond to definite events. By cascading the second function n times, the function

$\overline{B}(t+nd') = A(t) \qquad nd' \geq K + 1$

can be synthesized. Substitution of $\overline{B}$ for D, and U for E, gives the function

$E(t+d+nd') = B(t+nd') \wedge U(t-1+nd')$

For $t \geq 0$, $U(t-1+nd') = 1$, so that $E(t+d+nd') = A(t)$, which is a delay of $d + nd'$.

On the other hand, let D = U and E = $\overline{B}$. This gives the function

$E(t+nd'+d) = U(t+nd') \wedge B(t-1+nd')$

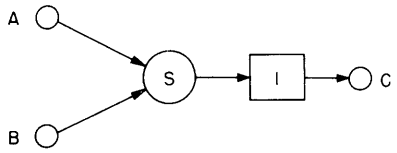
But for $t \geq 0$, we have $U(t+nd') = 1$, and hence $E(t+nd'+d) = A(t-1)$, which is a delay of $d + nd' + 1$, so that two relatively prime delays are available. [This proof of necessity is due to H. Loomis of M. I. T. A proof was also found by Dr. McNaughton, of the Moore School of Electrical Engineering, University of Pennsylvania, and by two students at the

1959 Summer Course at M. I. T.]

It should be noted that although all of the d_j may be quite large, the theorem does not require a reduction of the input frequency.

An example of a class of devices that satisfies condition (a) but not (b) (von Neumann[2]) is the class consisting of a single device – the Sheffer stroke with unit delay shown in Fig. IX-5.

For, given any net constructed entirely from Sheffer stroke devices with an arbitrary number of inputs, the output at time t + d can be affected only by those inputs at time t which have passed through d devices. But the Sheffer stroke function is monotonically



A(i)	B(i)	C(i+1)
0	0	1
0	1	1
1	0	1
1	1	0

Fig. IX-5. The Sheffer stroke with unit delay.

decreasing in that an increase of any of its arguments (in the ordering 1 > 0) cannot increase its value. It is easy to verify that if d is odd the output at time t + d must be a monotonically decreasing function of the inputs at time t and that if d is even the output at t + d must be a monotonically increasing function of the inputs at time t. Hence the function

$$C(t+d) = A(t) \wedge \overline{B(t)} + \overline{A(t)} \wedge B(t)$$

cannot be computed, since it is neither a monotonically increasing or monotonically decreasing function of its inputs. It is apparent that no reduction of the input frequency

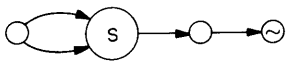


Fig. IX-6. Synthesis of a unit delay from the Sheffer-stroke device and an inverter.

will help in this case. However, the set of devices consisting of the Sheffer stroke with unit delay and an inversion with zero delay is complete, since a unit delay can be realized as shown in Fig. IX-6.

The inversion with delay zero is often accomplished by the use of the "two-line trick" in which every signal is carried on a pair of lines required to have complementary values. An inversion with zero delay is then realized by crossing the lines. This is

abstractly equivalent to the use of a two binary digit code for an input alphabet of two letters 0 and 1.

If the Sheffer stroke is regarded as an overly sophisticated example of an incomplete set of devices, it should be observed that an "and" gate, and "or" gate with delays of two, and an inverter with a delay of one are not complete for any input frequency by a similar argument.

As an example of synthesis in which delayed logic is used, suppose that the devices of Fig. IX-7a are available and that we want an elementary machine, with a single binary input A, which represents the definite event $A_{i-1} \wedge \overline{A_i} + \overline{A_{i-1}} \wedge A_i$ with some delay.

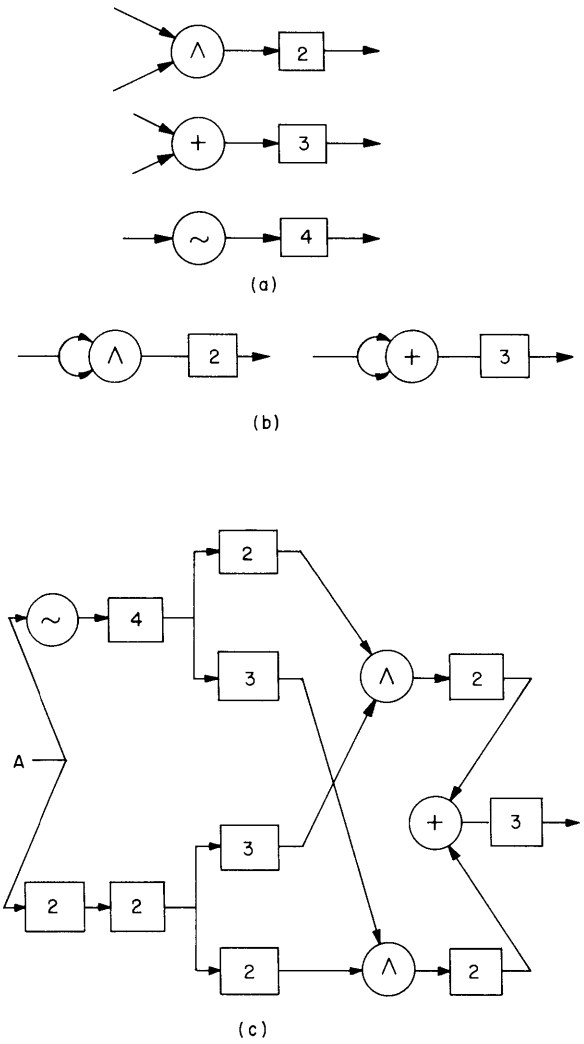


Fig. IX-7. (a) A complete set of devices. (b) Synthesis of 2-unit and 3-unit delays from the devices of (a). (c) Synthesis of $A_{i-1} \wedge \overline{A_i} + \overline{A_{i-1}} \wedge A_i$ from the devices of (a).

Since delays of 2 and 3 can be synthesized as shown in Fig. IX-7b, the synthesis should be possible, and indeed the circuit of Fig. IX-7c gives a representation with a delay of 11.

The following theorem, stated without proof, shows that the class of circle-free machines and the class of elementary machines are closely related.

THEOREM 2. Given a complete class of elementary devices, every circle-free machine represents a definite event and every definite event can be represented by a circle-free machine.

The synthesis argument of Theorem 1 yields circle-free machines, and since every backward path from the output of a circle-free machine must terminate in a finite number of steps at a direct input line, the first statement of the theorem is easily verified.

It should be emphasized that if a complete set of devices includes the constant Boolian functions 0 and 1, these are considered to be circle-free, although they may be diagrammed in terms of circles as shown in Fig. IX-8 which represents the definite event $t \geq 1$. In other words, circles that cannot be entered from a direct input line may

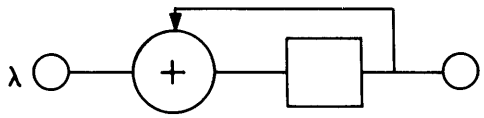


Fig. IX-8. The constant Boolian function 1.

be allowed. These devices represent definite events in the sense that the output is completely independent of past inputs at $t \geq 1$.

The machine of Fig. IX-9 represents the definite event Ω, where Ω denotes the empty set of input sequences. The set of all input sequences (including λ) is represented by the machine of Fig. IX-8.

The machines of Fig. IX-10 both represent the set of all sequences ending in 1, although one machine has a circle and one does not.

Kleene (1) has used a complete set of elements of the type shown in Fig. IX-11, which he has called "neurons." An output of 1 occurs one unit of time after at least h of the terminals $A_1, \ldots, A_n$ receive a one and none of the terminals $B_1, \ldots, B_m$ receive a one, and an output of one occurs only in this case. The terminals $B_1, \ldots, B_m$ are called inhibitory or inhibiting inputs and the $A_1, \ldots, A_n$ are called excitatory or exciting inputs. These are elementary machines, and it is easy to verify the fact that any definite event can be represented with delay 2 if its Boolian expression is put in normal disjunctive form (sum of products).

Nonelementary machines may occur in a complete set of devices. Minsky (8) has

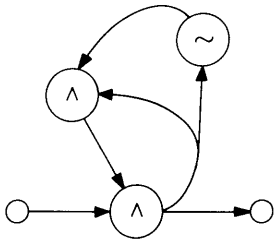


Fig. IX-9. A machine that represents Ω.

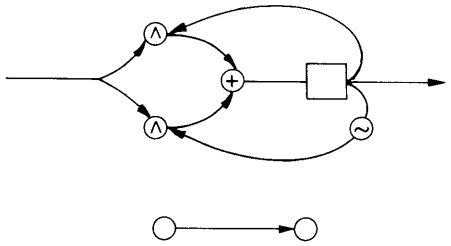


Fig. IX-10. Two machines which represent the set of all sequences terminating with a one.

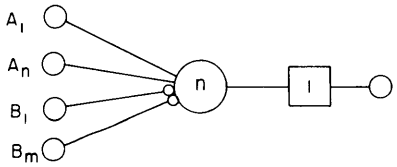


Fig. IX-11. A neuron with n excitatory and m inhibitory inputs.

observed that the device of Fig. IX-12a forms a complete set with $n \leq 4$. These devices "will not fire if they fired at the previous time." Such devices have been considered by McCulloch and Pitts as exhibiting some of the features of neurons. In terms of the abbreviated notation of Fig. IX-12a, an "and" gate with delay 3, "or" gate with delay 3, a delay of 3, and an inverter with delay 3 can be diagrammed as shown in Fig. IX-12b. The "clock" $t \equiv 2 \bmod 3$ can be generated as shown in Fig. IX-13a.

The devices of Fig. IX-12a are not elementary. For example, the device of Fig. IX-13b will have a nonzero output one unit of time after a sequence of an odd number of pairs of ones at its inputs, but not one unit of time after a sequence of an even number of pairs of ones. Consequently, the output cannot be determined by a

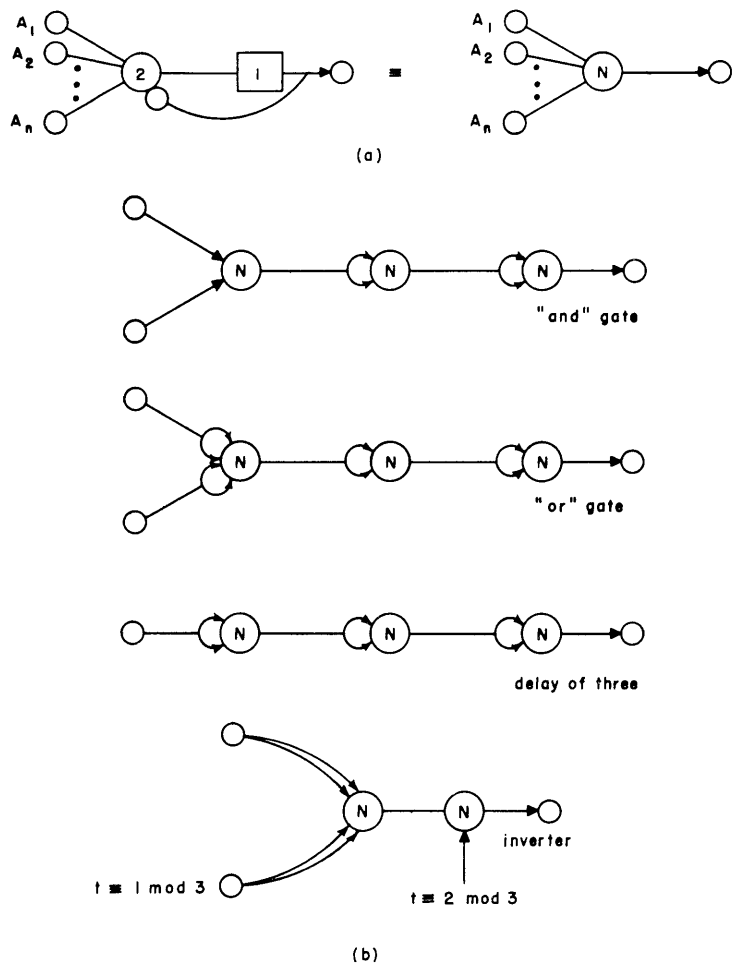


Fig. IX-12. (a) A non-elementary complete set of devices.
(b) The synthesis of a complete set of devices from the devices of (a).

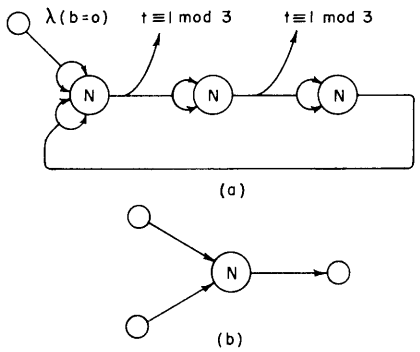


Fig. IX-13. (a) The generation of t ≡ 1, 2 mod 3.
(b) A non-elementary device.

bounded segment of the past. However, the devices of Fig. IX-12b are elementary, with inputs restricted to t = 3k, k ≥ 1.

3. Regular Events

Regular events are formed from finite events (which are, a fortiori, definite) by three algebraic operations called regular operations (labeled 1, 2, 3). Given the elementary machines that represent the finite events, these operations correspond to certain constructions of new machines that use the given elementary machines as building blocks. If the given elementary machines represent the corresponding finite events with delay zero, the construction is particularly simple. Since such representations are always

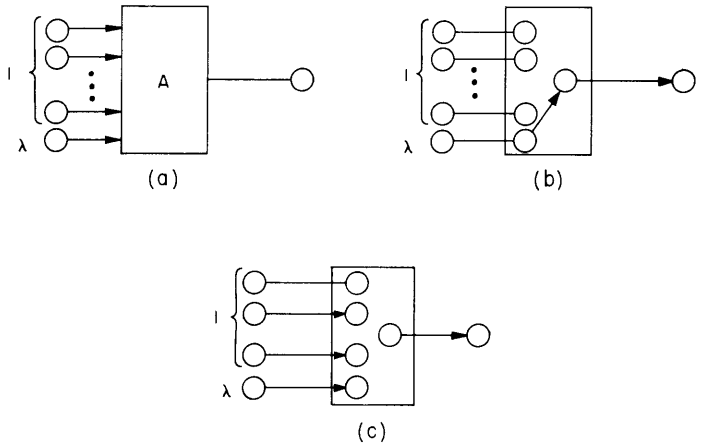


Fig. IX-14. (a) A machine that represents the set A of input sequences in the alphabet I. (b) A machine that represents the set Λ. (c) A machine that represents the empty set Ω.

possible by machines that use the complete set of devices of Fig. IX-2, called "instantaneous logic" in the preceding section, it will be assumed initially that such devices are available. The following constructions are similar to those used by Copi, Elgot, and Wright (4).

Given an event A (not necessarily finite), the machine that represents it will be diagrammed as shown in Fig. IX-14a. The bracketed inputs are direct inputs, and I denotes the set of input symbols. An output will occur at the same time as the last symbol of an input sequence of A, provided that the sequence is immediately preceded by an input of 1 at the bottom input terminal. If the "starting pulse" λ is applied to this input, the output therefore represents the finite event A. (Capital letters will be used to denote events.) The set of input sequences consisting only of λ will be denoted by Λ, and represented by the machine of Fig. IX-14b. The empty set of sequences or the impossible

event will be denoted by Ω and represented by the machine of Fig. IX-14c.

If A and B are two events, the set of input sequences represented by the machine of Fig. IX-15a will be denoted by A + B and is obviously the union of the sets A and B. In subsequent constructions of this type, it will be assumed that direct inputs to all component machines are returned to common direct input terminals for the composite machine. If A and B are two events, the set of input sequences represented by the machine of Fig. IX-15b will be denoted by AB and consist of all sequences formed by a sequence of A, immediately followed by a sequence of B. If A is an event, the set of all sequences represented by the machine of Fig. IX-16 will be denoted by A^*. It consists of the sequence λ and all sequences formed by the juxtaposition of any finite number of sequences in A. The set A^* can be formally expanded as

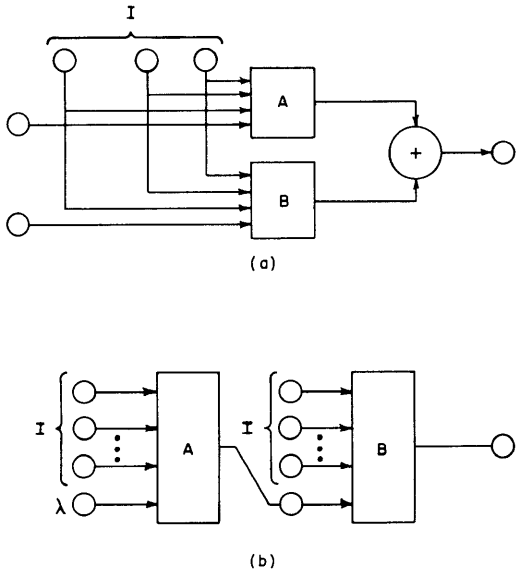


Fig. IX-15. (a) The construction of the machine that represents the set A + B from the machines that represent A and B. (b) The construction of the machine that represents AB from the machines representing A and B.

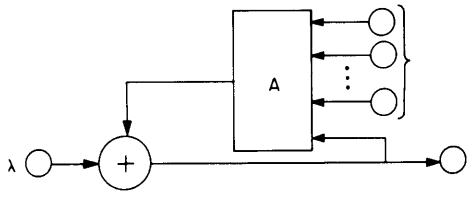


Fig. IX-16. The construction of the machine that represents A^* from the machine that represents A.

$$A^* = \Lambda + A + A^2 + A^3 + \ldots + A^n + \ldots$$

where A^n is an abbreviation for $\overbrace{AA\ldots A}^{n}$.

Any formal similarity to the formula for gain in the presence of a feedback loop is not coincidental. Servo engineers will note that if A were a real number less than one, this series would sum to $1/(1-A)$. Any expression formed from letters denoting finite events by using the regular operations a finite number of times will be called a regular expression, and the corresponding set of input sequences will be called a regular set or regular event. The finite events represented by the letters occurring in a regular expression will be called units, and the expression will be said to be a regular expression in terms of the pertinent units. It is easy to verify the fact that the addition defined above is commutative and associative and that the multiplication is associative and distributive over addition. The set Ω is an additive unit, and Λ is a left and right multiplicative unit.

It is apparent from the preceding considerations that any regular event can be represented by a finite machine if the complete set of devices which has been called "instantaneous logic" is available for the construction of the machine. It is of interest to inquire whether this is also true for other complete sets of elements, particularly those with unavoidable delays. In the first section of this report, we gave conditions on a complete set of elements which insured that any definite event could be represented with some delay at maximum input frequency. The question as to whether any regular event can be represented under these conditions was first raised by Kleene (1). The question can be refined by adding the requirement that the input frequency not be reduced.

The following example may help to clarify the question. The output of the circuit of Fig. IX-17a is described by $A_{i+1} = A_i \wedge \overline{B}_i + \overline{A}_i \wedge B_i$. If, at t = 1, the machine is started with no output at A, the event consisting of all input sequences that contain an odd

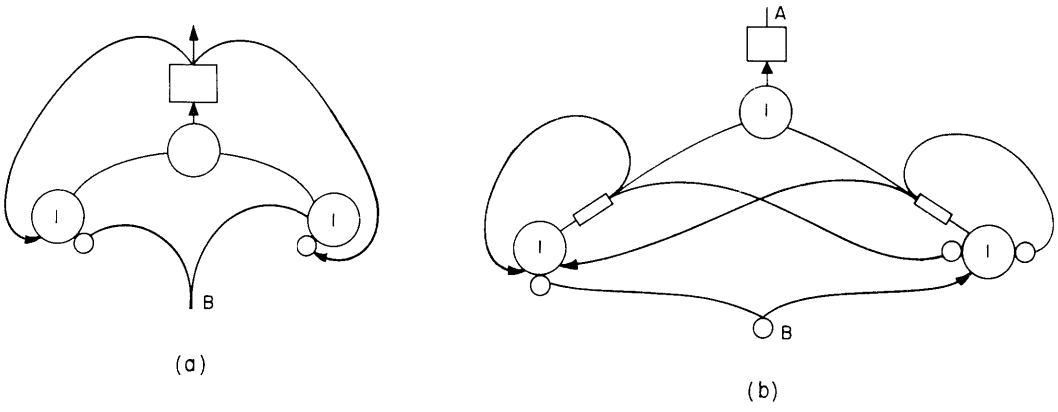


Fig. IX-17. (a) A neuron flip-flop. (b) A neuron flip-flop constructed from devices with delay.

number of ones with a delay of one is represented. This machine is sometimes described as a one-stage binary counter or a triggering flip-flop. If all of the logical elements have a delay of one, it is easily verified that the circuit will not represent the same event, even with a delay of two if the inputs are separated by a single clock interval.

For example, the input sequence 110000... will give the output sequence 001100..., although the correct result with a delay of two is 001000... . The delay around the feedback loop is too long for the effect of the first one to prevent the second one from reaching the output. A common solution to this problem is to require two clock intervals between inputs. However, the circuit of Fig. IX-17b represents the required event with a delay of two, without the necessity for reducing the input frequency. The difficulty is that more complex devices have been used.

The following theorem is a generalization of a theorem of Kleene (1).

THEOREM 3. If every definite event can be represented with a delay of S, with some complete set of devices used, then every regular event can be represented with a delay of S.

The difficulty with this result is that such complete sets consist of an infinite number of distinct types of devices, and it is unsatisfactory to require an infinite variety of packages in order to guarantee the representability of any regular event.

The proof of Theorem 3 rests on the following lemma.

LEMMA 1. Every regular event can be decomposed into the sum of two parts, one of the first kind, which is finite and contains only input sequences of length, at most, S - 1 and one of the second kind, which is a regular expression in terms of certain finite events called "units" that contain no input sequences of length less than S.

Any regular expression that can be decomposed according to the requirements of Lemma 1 will be said to be S-decomposable, and in the form required by Lemma 1 it will be said to be S-decomposed with that form as an S-decomposition.

PROOF. The result is trivial for any finite event. The sequences are divided into those of length less than S and those of length S, or greater. The second part is regular, since any finite set of sequences is regular, and can be considered as a unit.

Let A and B be two regular events (not necessarily finite) that can be decomposed into sums A' and A" and B' and B", respectively, where A', B' consist of sequences of length less than S, and A", B" are regular expressions in terms of finite events consisting of sequences of length at least S. Then A + B can be so decomposed as A + B = (A'+B') + (A"+B"), where the second term is regular in terms of units formed from sequences of length, at least, S. Hence any finite sum of decomposable events can be decomposed.

Similarly, if A and B are decomposable as described above, then AB = (A'+A")(B'+B") = A'B' + A"B" + A'B" + A"B". Since A'B' is finite, it can be decomposed by the procedure described above for finite events. The term A"B" is of the

second kind, since A" and B" are of the second kind.

The factor A" is, by hypothesis, a regular expression in terms of units consisting of sequences of length, at least, s. If A" is of the form $H_1 + H_2 + \ldots + H_n$, then A"B' is of the form $H_1B' + H_2B' + \ldots + H_nB'$.

A typical term in this sum, say H_1B', is then of the form $F_1F_2 \ldots F_mB'$. If F_m is a unit, it contains no sequences of length less than S, since A" was of the second kind. Then $F = F_mB'$ is a unit containing no sequences of length less than s, and $F_1F_2 \ldots F_{m-1}F$ is of the second kind. If $F_m = K^*$, then $F_mB' = K^*B' = B" + K^*KB'$, and $F_1F_2 \ldots F_mB'$ becomes

$$F_1F_2 \ldots F_{m-1}B' + F_1F_2 \ldots F_{m-1}K^*KB'$$

If K is a unit, it contains no sequences of length less than S, and $G = KB'$ is a unit containing no sequences of length less than S. Under these circumstances, $F_1F_2 \ldots F_mB'$ becomes

$$F_1F_2 \ldots F_{m-1}B' + F_1F_2 \ldots F_{m-1}K^*G$$

and the second term is of the second kind, and $F_{m-1}B'$ is treated by the methods described above. If K is not a unit, KB' must be similarly decomposed. Finally, if F_m is neither a unit nor of the form K^*, F_mB' must be decomposed in the same fashion.

By this recursive procedure, the term A"B' can be reduced either to B' + L or M, where L and M are of the second kind, since the sum of any finite number of terms of the second kind is of the second kind, and the sum of any finite number of terms of the first kind is of the first kind.

The term B'A" can be treated in an exactly analogous manner, and hence A'B' + A'B" + A"B' + A"B" can be reduced to the sum of a term of the first kind and a term of the second kind.

By an obvious induction, any product of s-decomposable terms can be s-decomposed, and hence any sum of products of s-decomposable terms can be s-decomposed.

Finally, if A is s-decomposable, then a term of the form A^* is also s-decomposable. First, if $A \supset \Lambda$, then $A^* = (\Lambda+B)^* = \Lambda + B^*$, where B does not contain Λ, and its shortest sequence is of length, at least, 1. Then $\Lambda + B^* = \Lambda + (\Lambda+B+\ldots+B^{s-1})(B^s)^*$. If A is s-decomposable, so is B, and hence $(\Lambda+B+\ldots+B^{s-1})$ can be decomposed by repeated applications of the methods described above. If this decomposition is given by $A + B + \ldots + B^{s-1} = C' + C"$, and $B^s = D$, then A^* becomes $A + (C'+C")D^* = \Lambda + C'D^* + C"D^*$.

If D is finite, it is a unit of the second kind, and hence $C"D^*$ is of the second, and $C'D^*$ must be recursively decomposed as the product of terms of the first and second kind. If D is not finite, it must itself be decomposed, but the decomposition will yield no terms of the first kind, since D contains no sequences of length shorter than S. After

the completion of the decomposition of D, $C''D^*$ will be of the second kind, and $C'D^*$ must be decomposed as a product of terms of the first and second kinds. This recursion completes the proof of Kleene's lemma.

LEMMA 2. If a machine represents an event with any delay d, there is a Boolian function of the direct inputs at time t, and the inputs to all delay elements at times t - 1, t - 2, ..., t - k, where k is the length of the longest delay in the machine.

Since inputs after time t cannot affect the output at time t + d and the machine is deterministic, Lemma 2 must be true.

The proof of Kleene's theorem now follows from three constructions.

(a) If A and B denote two regular events, then the machine that represents A + B with delay s can be constructed as follows. If f_A and f_B are the Boolian functions of Lemma 2 associated with the machines that represent A and B, then the Boolian function $f_A + f_B$ can be computed with delay s, since the function describes a definite event with a rather large input alphabet. This arrangement can be described by Fig. IX-18a.

(b) Let A and B be two regular events of the second kind. Then the net of Fig. IX-18b

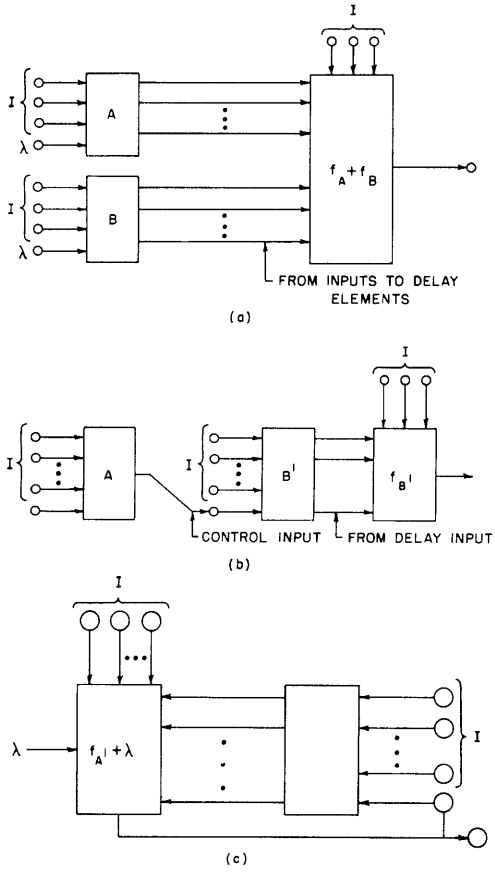


Fig. IX-18. (a) The computation of A + B from A and B. (b) The computation of AB from A and B. (c) The computation of A^* from A'.

represents AB. The machine B' represents B with the restriction that the control input is required S units of time after the first symbol in a sequence of B. Since no sequence of B has length greater than S, this gives a representation of AB. As shown above, it can be assumed that $f_{B'}$ is calculated with delay s. Events of the first kind need not be considered, since by the s-decomposition lemma they can be represented by a single elementary machine because in their totality they constitute a finite event, and the sum of this result and the event represented by the term of the second kind can be represented by the technique of construction (1).

(c) If A is a regular event of the second kind, A^* can be represented by the arrangement of Fig. IX-18c. Again, $F_{A'} + \lambda$ is computed with delay s, and A' represents A, with the restriction that the initiating pulse is required at the time of the s^{th} symbol in a sequence of A. Since no sequence of A has length less than s, no sequence of A can be overlooked by this requirement. A simple induction completes the proof of Kleene's theorem.

4. Abstract Machines

The abstract description of a machine is based on the notion of the state of a machine.

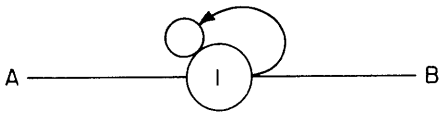


Fig. IX-19. Improper circle.

Any description of the total state of a machine must completely determine the inputs to the machine and outputs of every device in the machine. This implies that machines for which certain outputs are not well defined must be excluded. For example, in the circuit of Fig. IX-19 if there is an input at A, the output at B is not well defined. It is left for the reader to verify the fact that if every circle contains a delay, such difficulties cannot occur. This requirement will be assumed to be satisfied by all machines discussed here.

Since every circle contains a delay element, the outputs of all of the devices in a given machine are uniquely determined by the direct inputs to the machine and by the outputs of the delay elements. This can be verified by following all paths terminating at a given device backward until either a delay element or direct input line is reached. An unambiguous description of the outputs of the delay elements determines the internal state of the machine, and a specification of each input determines the input state. It should be noted that for these definitions, an n-unit delay must be considered as n separate unit delays. A specification of both the input and internal state determines the total or complete state of a machine.

In a synchronous machine it is assumed that the state is well-defined only at clock times and that more or less complicated transients occur in the clock interval between clock times.

Abstractly, the behavior of a finite-state machine is most conveniently described by a directed graph, with internal states represented as nodes and the effect of inputs as transitions between internal states as directed branches connecting the corresponding nodes. As an example, the circuit of Fig. IX-20 has two internal states, 1 and 2, corresponding to the two possible outputs, 0 and 1, respectively, of the delay element, and four total states, since only two input states are presumed to be possible. The behavior of the machine can be described by the graph shown in Fig. IX-21.

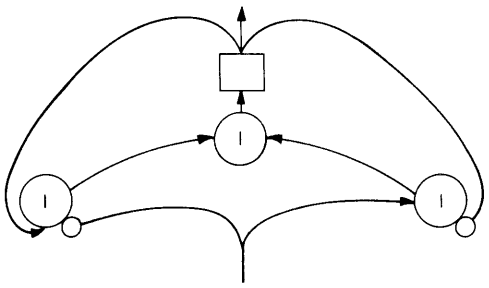


Fig. IX-20. A one-stage binary counter.

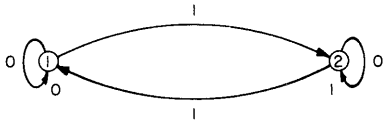


Fig. IX-21. A simple computer and its state diagram.

If the left node represents a zero output and the right node a one output from the delay element, and the numbers on the directed branches denote inputs, this diagram is an abstract representation of the behavior of the preceding circuit. The output of this particular circuit depends only on the internal state, but it is clear that it may, in general, depend also on the input. For this reason, the output is generally noted on the branch corresponding to the input on which it depends. The complete internal state transition diagram for this example is shown in Fig. IX-22.

For a synchronous machine, a circle and the symbols on a directed line leading from it can be interpreted as the internal state, input and output at time t, and the circle at the termination of the line as the state at the next clock time. This situation can be most graphically described if the machine is represented in the block diagram form of Fig. IX-23. No delays occur in the block marked "logic." The outputs of the unit delay elements determine the "present" internal state, and the inputs determine the "next" internal state. The outputs are logical functions of both the input and "present" internal state.

A total-state transition diagram can be derived by assigning a state to every transition of the internal-state diagram and making appropriate transition connections. If

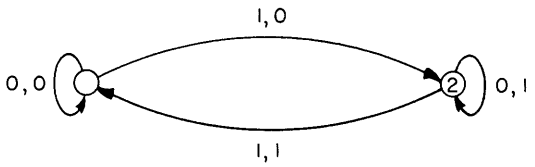


Fig. IX-22. Internal-state transition diagram with outputs.

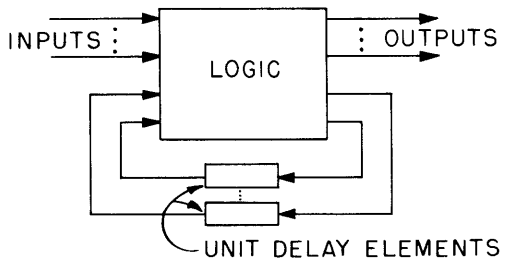


Fig. IX-23. Block diagram of machine.

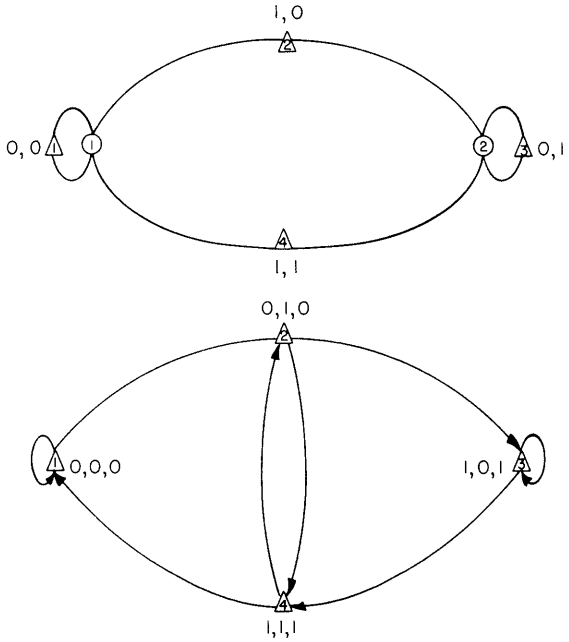


Fig. IX-24. The generation of a total-state diagram.

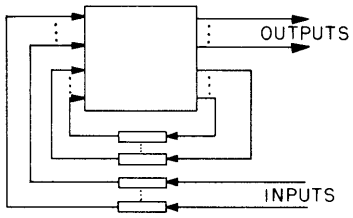


Fig. IX-25. The inclusion of the inputs in the total state.

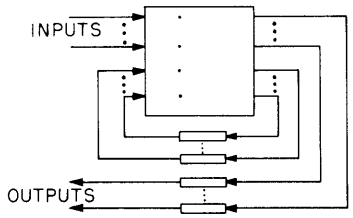


Fig. IX-26. The inclusion of the outputs in the total state.

triangles are used to denote the total states, the procedure can be carried out in the steps that are illustrated in the example of Fig. IX-24. Each triangle is associated with the input and output of its branch and the internal state corresponding to the preceding circle. The circles are then removed from the diagram by connecting each triangle to those triangles that can be reached by passing through, at most, one circle. For the example, this procedure yields the result shown in the second step of Fig. IX-24, in which the numbers associated with each total state determine internal state, input, and output in that order. Each total state has two output lines that go to total states corresponding to the two possible input states, and the lines leading from any state could be labeled accordingly.

The total state is determined by all lines entering the logic block of the block diagram. This can be more vividly illustrated if the inputs are delayed as shown in Fig. IX-25. The outputs of the delays then determine the "present" total state and the delay inputs determine the "next" total state. The output is a logical function of the "present" total state.

An alternative construction is sometimes used to insure that the outputs depend only on the state of the machine. This construction is described by the block diagram of Fig. IX-26, in which, again, the output of the delay elements determines the "present" state, and the inputs, the "next" state. It is left for the reader to determine how the state diagram for this configuration is derived from the internal-state diagram with the delays in the output lines omitted.

While the notion of total state is probably most basic, the internal-state diagram has fewer nodes and is most often used. Consequently, a "state diagram" will be assumed to imply "internal-state diagram," unless specific exception is made.

Given a state diagram, an immediate synthesis of a corresponding machine is possible by replacing each node by an "or" gate with output entering a delay element. For any given state there is an input to the corresponding "or" gate associated with each transition leading to the state that is the appropriate Boolian function of the input and the output of the delay element corresponding to the state at which the transition originates.

Circuits for the computation of these functions then yield appropriate inputs for the corresponding delay elements. A simple state diagram and its synthesis by this method are shown in Fig. IX-27. The information contained in a state diagram can be represented in many ways. The following representation is particularly interesting in view of its analytical possibilities. A matrix in which the element A_{ij} is the set of input symbols that cause a transition from state i to state j in some ordering of the states of a machine will be called a "state-transition matrix." It is apparent that the collection of sets described by a row of such a matrix constitutes a partition of the input alphabet if, for every state, a unique next state is defined for every symbol of the input alphabet.

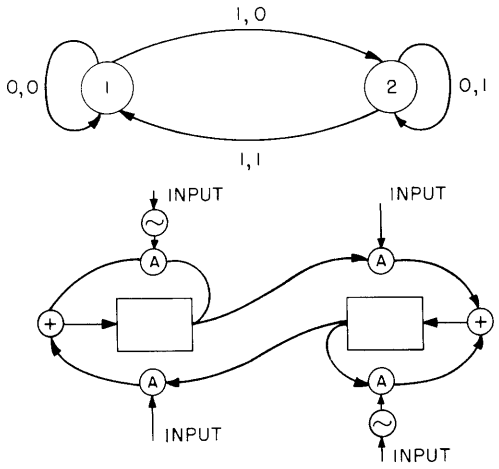


Fig. IX-27. An example of the "direct" synthesis of a machine from its state diagram.

It will be assumed that the output depends only on the state. This is no restriction of generality, since it is always true for total-state diagrams, and can be achieved for internal high state diagrams by splitting every state into two states with identical exit transitions, one of which is entered by all transitions with output one, and the other is entered by all transitions with output zero. This amounts to placing a delay unit in the output line and including it in the definition of the internal state. Our example can be treated directly because the output is a function only of the state.

In the following discussion let U denote the set consisting of the single input symbol one, let Z denote the set consisting of the single input symbol zero, and let Λ indicate that the machine is placed in state 1 at t = 0.

Now if X_1 denotes the class of input sequences that leave the machine in state 1, and Y_1 denotes the class of input sequences that leave the machine in state 2, the following set of equations is valid in the regular algebra.

$$\Lambda + X_1 Z + Y_1 U = X_1$$

$$\overline{X}_1 U + \overline{Y}_1 Z = \overline{Y}_1$$

In matrix notation this becomes

$$(\Lambda, 0) + (X_1, Y_1) \begin{pmatrix} Z & U \\ U & Z \end{pmatrix} = (X_1, Y_1)$$

Similarly, if the machine is started in state 2, the sets of input sequences X_2 and Y_2 which leave the machine in states 1 and 2, respectively, are solutions of the system.

$$(0, \Lambda) + (X_2, Y_2) \begin{pmatrix} Z & U \\ U & Z \end{pmatrix} = (X_2, Y_2)$$

These equations can be solved algebraically. The solution can be carried out recursively if the equation $XA + B = X$ can be solved.

THEOREM 4. The set of sequences satisfying the equation $XA + B = X$ is unique and described by $\underset{\sim}{X} = BA^*$ if and only if $A \not\supset \Lambda$.

PROOF. Any solution clearly contains BA^*. Suppose $BA^* + C$ is a solution with $C \wedge BA^* = 0$. Then, by substitution, we have $BA^* + CA = BA^* + C$. If both sides of this equation are intersected with C, then $CA \wedge C = C$ or, equivalently, $C \subset CA$. But the shortest sequence of C is shorter than the shortest sequence of CA, contradicting $C \subset CA$, unless C is empty.

If $A \supset \Lambda$, then the set $(B+C)\,A^*$ is a solution for any C which $B \subset C = 0$. By substitution

$$(B+C)\,A^*A + B = (B+C)\,A^*$$

$$BA^* + CA^*A = BA^* + CA^*$$

But since $\Lambda + A = A$, we have

$$BA^* + CA^*(\Lambda+A) = BA^* + CA^*$$

$$BA^* + CA^* + CA^*A = BA^* + CA^*$$

and since $CA^*A \subset CA^*$, we obtain

$$BA^* + CA^* = BA^* + CA^*$$

By using this result, the solution to

$$\Lambda + X_1Z + Y_1U = X_1$$

$$X_1U + Y_1Z = Y_1$$

can be found in the following way: Consider

$$X_1Z + (\Lambda+Y_1U) = X_1$$

The solution for X_1 is $X_1 = (\Lambda+Y_1U)\,Z^*$. Substituting in the second equation and collecting terms, we obtain

$$Y_1(UZ^*U+Z) + Z^*U = Y_1$$

which can be solved for Y_1, and yields

$$Y_1 = Z^*U(UZ^*U+Z)^*$$

Substituting this result in the solution for X_1 in terms of Y_1 gives

$$X_1 + Z^* + Z^*U(UZ^*U+Z)^*\,UZ^*$$

This procedure obviously generalizes, and it is easy to see that at no step does the coefficient of the variable which is being solved for contain Λ. Indeed, since none of the coefficients contain λ (a transition can only be caused by some member of the input alphabet), any substitution for any variable results in an expression that has some member of the input alphabet as a factor and hence does not contain Λ.

Solutions to equations of this type can also be found directly from a state diagram by the methods of flow-graph analysis. For example, the diagram of Fig. IX-28 is derived from the state diagram of a one-stage counter by splitting node 2. The class

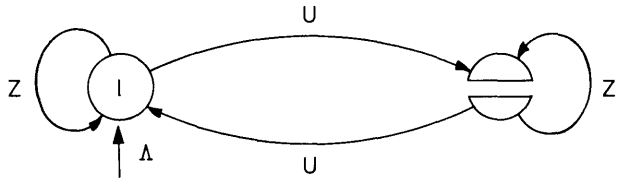


Fig. IX-28. Illustration of a "split" node.

of all sequences that arrive at node 2 for the first time, starting from node 1, is given by $\Lambda Z^*U = Z^*U$, where Z^* is analogous to the loop "gain" around node 1. The loop "gain" around node 2 is $(UZ^*U+Z)^*$, where the two terms in parentheses correspond to the two loops around node 2. The product of these "gains" gives $Z^*U(UZ^*U+Z)^* = Y_1$ for the "gain" from node 1 to node 2, which is the same as the result for Y_1 derived analytically. However, the "gain" around node 1 is easily seen to be $(Z+UZ^*U)^*$, which is not the same as the earlier result, but is easily seen by inspection to represent the same family of input sequences – namely, all sequences with an even number of ones. Very little is known about algebraic methods of recognizing such equivalences. The original set of equations can be solved for the variables in the reverse order; this procedure yields still another "equivalent" algebraic form for the result. Such difficulties are typical of noncommutative algebras, and it would be of great interest to have a complete set of identities for this algebra. If T denotes the matrix $\begin{pmatrix} Z & U \\ U & Z \end{pmatrix}$, and E denotes the "unit" matrix $\begin{pmatrix} \Lambda & 0 \\ 0 & \Lambda \end{pmatrix}$, then the matrix of solutions $\begin{pmatrix} X_1 & Y_1 \\ X_2 & Y_2 \end{pmatrix}$ will be denoted by T^* and it satisfies the matrix equation $E + T^*T = T^*E$. This is analogous to an equation of the form $E = T^*(E-T)$, so that T^* is analogous to $(E-T)^{-1}$ in the same way as A^* was analogous to $(1-A)^{-1}$ for an event A.

A generalization of this procedure demonstrates that for a given starting state, the class of all input sequences causing a machine to enter or return to a given state is a regular class. These classes are called state equivalence classes, and the procedure outlined above provides a straightforward method for the calculation of a regular

expression for these classes. If a subset of the states of a machine is associated with an output one and its complement with the output zero, then the set of all input sequences yielding an output one at their termination is the union of a finite number of state equivalence classes and hence is a regular set.

It has been shown that any regular set can be represented, at least with instantaneous logic, and in fact, with somewhat more general components. Since any finite machine can be represented by a state diagram, and the procedure above uses only regular operations, the following theorem of Kleene has been demonstrated.

THEOREM 5. A set of input sequences can be represented by an arbitrary finite-state machine only if it has a regular expression.

5. The Synthesis of Machines from a Complete Set of Devices

As the previous results show, the set of all input sequences causing a transition from the i^{th} to the j^{th} state of a sequential machine with transition matrix T is the element in the i^{th} row and j^{th} column of the matrix which is a solution of $XT + E = X$, where E is a matrix with Λ diagonal elements Λ and all other elements equal to Ω, the empty set of input sequences. The equation

$$XU \wedge U^* T + E = X$$

has the same solution, where U denotes a matrix with diagonal elements I and all other elements equal to Ω, and $\wedge$ indicates that the multiplicative operation in the matrix multiplication is not conjunction. The element $(U^*T)_{ij}$ in the i^{th} row and j^{th} column of U^*T is the set of all sequences terminating in an element of $(T)_{ij}$ (an indefinite event in the terminology of Kleene). The element $(XU)_{ij}$ is the set of all sequences formed from $(X)_{ij}$ by the addition of an arbitrary element of the input alphabet.

Alternatively, it is easily seen that T^* also satisfies

$$X \cdot T^k + (E+T+T^2+\ldots+T^{k-1}) = X$$

or, equivalently,

$$XU^k \wedge U^* T^k + (E+T+T^2+\ldots+T^{k-1}) = X$$

for any positive integral k.

In this expression, the computation required by the operations $\wedge$ and + depends only on the dimensions of the operand matrices which are determined by the number of states of the machine. Let the delay required for the computation of the product be d_a, and the delay required for the computation of the sum be d_s. No circles are involved in these computations, so they can be accomplished with a complete set of devices.

The events U^*T^k are definite and the events $(E+T+\ldots+T^{k-1})$ are finite. Hence these

events can be computed by a complete set of devices. If two relatively prime delays are available, then U^*T^k and $E + T + \ldots + T^{k-1}$ can be computed with delay d_x and $d_x + d_a$ respectively, where d_x may depend on k.

Let $k = d_a + d_s$, and let d_x be chosen accordingly. Then

$$XU^{d_a+d_s} \wedge U^*T^{d_a+d_s} + E+T+\ldots+T^{d_a+d_s-1} = X$$

Multiplying both sides on the right by $U^{d_a+d_s+d_x}$ and recombining gives

$$XU^{d_a+d_s+d_x} \wedge U^*T^{d_a+d_s}U^{d_x}U^{d_a} + E+T+\ldots+T^{d_a+d_s-1}U^{d_x+d_s}U^{d_s} = XU^{d_a+d_s+d_x}$$

This equation is a prescription for the construction of a machine that represents

$$XU^{d_a+d_s+d_x}$$

or, alternatively, which represents X with delay $d_a + d_s + d_x$.

This construction is independent of the initial state. For a specific initial state, the equations $XT^k + (E+T+\ldots+T^{k-1}) = X$ may be multiplied on the left by a row vector S_o whose only nonempty component is a Λ in the position corresponding to the initial state. A block diagram of the synthesis for this case is given in Fig. IX-29. In any case, the n^2 elements of $U^*T^{d_a d_s}$ must be computed, where n is the number of states. Each

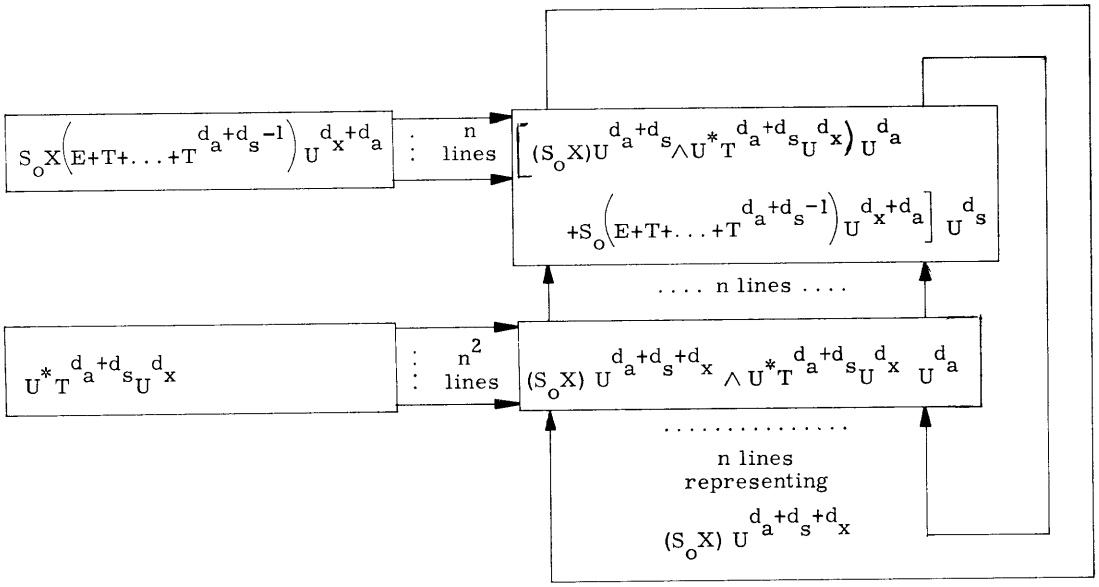


Fig. IX-29. Representation of IT^* in terms of the elements A_{ij} of T^p and the elements B_k of $I(E+T+T^2+\ldots+T^p)$.

binary output that is a function of the state may, of course, be computed (with some additional delay) by a circle-free net. This is denoted algebraically by multiplication of $XU^{d_a+d_s+d_x}$ on the right by a column vector with nonempty components Λ only in those positions corresponding to an output of 1.

6. Conclusions

The delays discussed here may, in practise, be due to lengths of wire, strip line, coaxial cable, or devices required to restore rise times. The construction of Fig. IX-29 makes it clear that a machine may be fast, even when arbitrarily long delays are attached to the output of every logical device – a rather extreme situation – provided only that they are accurately known and that delays of relatively prime numbers of clock intervals can be constructed.

The price of such delays is not a slow machine but a relatively large wait between inputs and results. In many applications this will be unimportant, and it is tempting to guess that even when the computer itself is in a feedback loop, an analysis of the overall system could remedy any difficulty caused by the computer's delay in much the same way as delays in the feedback paths inside the computer itself can be handled.

Another result of such delays is an increased number of devices. It might be argued that this increase, in turn, would make the required interconnections longer in a self-defeating way. A careful inspection of the last construction shows that U^*T^k and $U^*T^{d_a+d_s}$ and $E + T + \ldots + T^{d_a+d_s-1}$, being circle-free, can be laid out over large areas in several cities remote from the circuitry involving the state feedback lines, provided only that their output signals arrive at this circuitry at the proper times. The compilation of the feedback circuitry depends only on the number of states and not on the particular computation being performed.

D. N. Arden

References

1. S. C. Kleene, Representation of events in nerve nets and finite automata, Automata Studies, Annals of Mathematics Studies No. 34, edited by C. E. Shannon and J. McCarthy (Princeton University Press, Princeton, N. J., 1956), pp. 3-42.

2. J. von Neumann, Probabilistic logics and the synthesis of reliable organisms from unreliable components, Automata Studies, Annals of Mathematics Studies No. 34, op. cit., pp. 43-98.

3. E. F. Moore, Gedanken experiments on sequential machines, Automata Studies, Annals of Mathematics Studies No. 34, op. cit., pp. 129-156.

4. I. M. Copi, C. Elgot, and J. B. Wright, Realization of events by logical nets, J. Assoc. Computing Machinery 5, 181-196 (April 1958).

(References continued on following page)

5. M. O. Rabin and D. Scott, Finite automata and their decision problems, IBM J. Res. Develop. 3, 114-125 (April 1959).

6. J. C. Shepherdson, The reduction of two-way automata to one-way automata, IBM J. Res. Develop. 3, 198-200 (April 1959).

7. R. C. Jeffrey, Some Recent Simplifications of the Theory of Finite Automata, Technical Report 219, Research Laboratory of Electronics, M. I. T., May 27, 1959.

8. M. L. Minsky, Some universal elements for finite automata, Automata Studies, Annals of Mathematics No. 34, op. cit., pp. 117-128.