# CANONICAL REGULAR EXPRESSIONS AND MINIMAL STATE GRAPHS FOR DEFINITE EVENTS*

J. A. Brzozowski**
Department of Electrical Engineering, Princeton University

This paper considers the class of sequential circuits which can be realized with unit delays and combinational circuitry, but no feedback loops. In the language of regular expressions, these circuits realize definite events. A theorem is proved that for every definite event there exists a canonical regular expression. An algorithm is presented for obtaining a minimal state graph directly from the canonical form of the regular expression. Methods for testing whether a regular expression is definite are discussed, and a procedure is described for converting a regular expression denoting a definite event to the canonical form. It is shown that canonical forms can be found for certain other classes of regular expressions, and that minimal state diagrams can be obtained directly from these canonical forms.

## I. INTRODUCTION

We are considering a class of sequential circuits which can be constructed without the use of feedback loops. Such circuits are said to realize definite events.

A brief review of the properties of finite automata and regular expressions is first presented, and definite events are defined. Next, we consider the class of non-initial definite events: A non-initial definite event consists of all sequences ending in a specified finite set of sequences. Canonical forms for non-initial definite events are discussed and methods are developed for constructing minimal state diagrams for such events. These methods are extended to the entire class of definite events and to certain other related events. Finally, we describe procedures for testing whether an event is definite and for converting regular expressions denoting definite events to canonical forms.

## II. FINITE AUTOMATA AND REGULAR EXPRESSIONS

A finite automaton is an abstract model of a sequential circuit. We are considering the class of fixed, finite automata, [1, 2] which are discrete, deterministic and synchronous. More specifically, a finite automaton has n binary inputs $x_1, x_2, \ldots, x_n$; m internal states

* This work was supported in part by the Bell Telephone Laboratories, Murray Hill, New Jersey.

** At present with the Department of Electrical Engineering, The University of Ottawa, Ottawa, Canada.

Presented at the *Symposium on Mathematical Theory of Automata*, Polytechnic Institute of Brooklyn, April 24, 25, 26, 1962

$q_1, q_2, \ldots, q_m$ (one of which, $q_\lambda$, is the starting state); and one binary output. The automaton is deterministic in the sense that, given the present state and the present input, the next state and the next output are uniquely determined. It is assumed that the history of an automaton can be described as having occurred at certain discrete moments of time, i. e., the automaton is synchronous. Without loss of generality, it is assumed that the intervals between these moments are all of equal length, and last for one unit of time, i. e., the history of the automaton takes place at times t = 1, 2, . . . , . At t = 0, before any input sequences are applied, the automaton is in its starting state, $q_\lambda$.

The class of sequential circuits, for which the above model is applicable, includes clocked sequential circuits and circuits operating in pulse mode.[3] Furthermore, a similar model can be used to represent the behavior of iterative combinational networks.[4]

For notational convenience, the $k = 2^n$ input n-tuples (of 0's and 1's) are coded into integers corresponding to the binary numbers formed by the input n-tuples. This allows us to represent the input state by a single $2^n$-valued input x. The integers, over which x ranges, are called input symbols, and the set of all input symbols is called the input alphabet, $A_k = \{0, 1, 2, \ldots, k-1\}$, where $k = 2^n$. For example, for k=2, the input n-tuples are coded: (0,0) -0; (0, 1) -1; (1, 0) -2 and (1, 1) -3.

The behavior of a finite automaton can be described by a flow table[5] or a state diagram (or state graph).[1,2] Two models are commonly used:

1) The Mealy model,[2] in which the output, z, is associated with transitions between internal states,

2) The Moore model,[1] in which the output, Z, is associated with an internal state.

Table I and Fig. 1 give examples of flow tables and state diagrams for the two models. In the Mealy model, the outputs may be interpreted as pulses occuring during the application of input pulses, whereas in the Moore model, the outputs may be considered as being level outputs, depending only on the state.

An input sequence, applied to a finite automaton in its starting state, is said to be accepted, if and only if the output corresponding to the last member of the sequence is 1. The behavior of an automaton can be completely described by specifying all the input sequences which are accepted (the remaining sequences being rejected). In general, the set of sequences accepted by an automaton is infinite, but can be represented by a finite formula, namely, by a regular expression.[6-13] Regular expression is defined recursively, as follows:

1. The symbols 0, 1, . . . , k-1, $\lambda$, $\phi$ are regular expressions.



TABLE I: EXAMPLES OF FLOW TABLES

(a) Flow Table for the State Graph of Fig. 1(a)

| Present State | x = 0 | x = 1 |
|---|---|---|
| A | A, 0 | B, 0 |
| B | B, 0 | A, 1 |

Next State, z

(b) Flow Table for the State Graph of Fig. 1(b)

| Present State | x = 0 | x = 1 | Z |
|---|---|---|---|
| A | A | B | 0 |
| B | A | B | 1 |

Next State

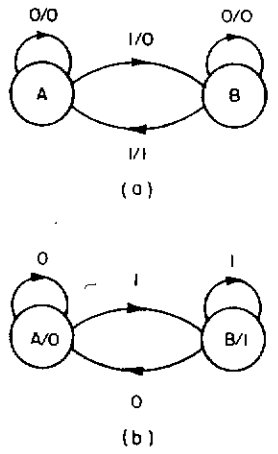


Fig. 1 Example of state graphs. (a) Mealy model; (b) Moore model.

2. If P and Q are regular expressions, then so are (P + Q), (P · Q) and P*.

3. Nothing else is a regular expression, unless its being so follows from a finite number of applications of rules 1 and 2.

The symbol $\lambda$ denotes the sequence of zero length and $\phi$ denotes the empty set of sequences.[11] More properly, $\lambda$ is the set of sequences consisting of a single member, the sequence of zero length. However, for economy of notation, the same symbol will be used to denote a sequence and the set consisting of only that sequence.[10,11] The operator (+) is called the sum or union; the operator ( · ) denotes product or concatenation [and is usually omitted, i. e., (P · Q) is written (PQ)]; the operator (*) is called the star operator.

Regular expressions are finite formulae denoting sets of sequences. If P and Q are two regular expressions denoting the sets E and F, respectively, then (P + Q) denotes the set E ∪ F (union of E and F); (P · Q) denotes the set of all sequences of the form, ef, $e \in E$, $f \in F$; and P* is the set defined by the infinite sum:

$$P^* = \lambda + P + PP + PPP + \ldots \quad (1)$$

The sum and product operations are associative; hence parentheses can be omitted. The sum is commutative but the product is not commutative.

The empty set $\phi$ has the properties of an additive and a multiplicative zero, i.e.,

$$\phi + R = R + \phi = R, \tag{2}$$

$$\phi R = R\phi = \phi. \tag{3}$$

The sequence of zero length has the property of a multiplicative unity,

$$\lambda R = R\lambda = R. \tag{4}$$

It has been shown by Kleene[6] that, for every regular expression R there is a finite automaton which accepts only the sequences denoted by R, and that the sequences accepted by any finite automaton can be denoted by a regular expression. Thus a regular expression contains just as much information as a flow table or a state diagram.[13] There are several ways for obtaining a regular expression from a state diagram.[9-11] In general, there are many regular expressions corresponding to the same state diagram, i.e., the same set of sequences can be described by many different expressions.

Two regular expressions P and Q are said to be "equivalent," P = Q, if and only if they denote the same set of sequences. A regular expression P is said to contain another regular expression Q, $P \supset Q$, if and only if the set denoted by Q is a subset of the set denoted by P. Clearly P = Q, if and only if $P \supset Q$ and $Q \supset P$.

## III. DEFINITE EVENTS

Sets of sequences applied to a finite automaton can be described in terms of events,[6-11] an event being a subset R of the set I of all possible input sequences. An event occurs when the input sequence is a member of the set of sequences constituting the event. We say that a finite automaton M realizes an event R, if and only if the set of sequences accepted by M is the set of sequences of R. By definition, an event is regular if and only if the set of sequences constituting the event can be described by a regular expression.

We shall now consider a certain restricted class of regular events, namely, definite events. To facilitate the discussion we define the symbol i to be:

$$i = 0 + 1 + \ldots + (k-1). \tag{5}$$

The event denoted by i consists of all the sequences of length 1, that is, all the input symbols. The set of all possible input sequences is then

$$I = i^* = \lambda + i + i^2 + \ldots, \tag{6}$$

where $i^2$ is a shorthand for ii. Thus $i^n$ denotes the set of all sequences of length n.



DEFINITION 1: An event is definite[6] if and only if it can be described by a regular expression of the form

$$R = E + i^*F, \tag{7}$$

where E and F are finite sets of finite sequences.

A definite event is called initial if and only if it can be described by a regular expression of the form R = E; it is called non-initial if and only if it can be denoted by a regular expression of the form $R = i^*F$. If a definite event is neither initial nor non-initial, it is called composite.

For example, the event $E = 3 + 012$ is an initial definite event, whereas $R = i^*(3 + 012)$ is a non-initial definite event, consisting of all sequences ending in either 3 or 012. The event $P = 0 + i^*11$ is a composite definite event.

It is clear from the definitions that a non-initial definite event $R = i^*F$ is an event consisting of all the sequences ending in a specified finite set of sequences F (including the sequences of F, since $i^*$ contains $\lambda$). If the longest sequence in F is of length q, then whether or not R has occurred at time t can always be determined by the inspection of at most the last q input symbols. On the other hand, in order to determine whether or not a composite or an initial event has occured at time t, in addition to knowing the last q input symbols (where q is the length of the longest sequence in the set E + F), we must also know the relation of t to the time origin. For example, if the event is E = 01 and the last two symbols of the input sequence are 01 at time t, it is not known whether E has occurred at time t, unless t is known; that is, E has occurred if t = 2, but not if $t \neq 2$.

It should be pointed out that a definite event may be denoted by a regular expression which does not have the form $R = E + i^*F$. For example, by inspection of the diagram of Fig. 1(b), we conclude that an output is produced for all sequences ending in a 1; that is, the event can be denoted by $R_1 = (0 + 1)^*1$. On the other hand, using signal flow graph techniques,[12] we obtain $R_2 = (0 + 11^*0)^*11^*$. The two expressions are of course equivalent, since they describe the same event, although the equivalence is not obvious. Methods for converting a given regular expression denoting a definite event to the form $R = E + i^*F$ will be discussed later.

Definite events can also be described by Boolean functions of the inputs.[6, 9, 11] For example, if x denotes the binary input, Z the output, and $R = 0 + i^*11$, we have

$$Z(1) = x'(1);$$

$$Z(t) = x(t-1) \,\&\, x(t), \quad t \geq 2;$$

where $x'$ denotes the complement of $x$ and $x \& y$ is the conjunction of $x$ and $y$. It follows that definite events can be realized without feedback loops, by storing the input signal for the last $q-1$ units of time. Methods of realizing definite events have been described elsewhere.[6, 9, 11]

Because of the simplicity of the form (7) we have chosen to define definite events in terms of regular expressions. Perles, Rabin, and Shamir[14] have used a slightly different approach, but they have shown that their definition is equivalent to definition (7). (See also Section X.)

## IV. CANONICAL FORMS FOR NON-INITIAL DEFINITE EVENTS

For the moment we shall restrict our attention to non-initial definite events, because they possess certain special properties; the methods discussed will be extended later to cover initial and composite definite events.

Sequences of input symbols can be arranged in increasing order. First, the sequences are arranged in order of increasing length; secondly, the sequences of length $q$ can be ordered lexicographically, say according to the size of the base 2 integer corresponding to a given sequence. For example, for $A_k = \{0, 1\}$, the sequences are arranged as follows:

$$\lambda,\ 0,\ 1,\ 00,\ 01,\ 10,\ 11,\ 000, \ldots$$

We shall use such an ordering throughout.

Given a definite expression (which we assume to be non-initial, unless otherwise specified), $R = i^*F$, let $f_j$ denote a sequence of $F$. Then we can write

$$R = i^*F = i^*(f_1 + f_2 + \ldots + f_m) = i^* \sum_{j=1}^{m} f_j . \tag{8}$$

The sequences $f_j$ will be called the (non-initial) terms of $R$. A term $f_j$ of $R$ is covered by another term $f_k$ if and only if $f_j = sf_k$, where $s$ is a finite sequence (possible of zero length). In other words, if $f_j$ ends in $f_k$, then $f_j$ is covered by $f_k$. Clearly, the covered terms can be removed from $R$ without changing the set of sequences denoted by $R$. This follows because $i^*f_k$ contains all sequences ending in $f_k$ and, in particular, all sequences ending in $sf_k = f_j$.

A definite expression in the form $R = i^*F$ is said to be in a definite (non-initial) form. It is said to be in a canonical form, if and only if:

1. $R$ is in definite form.
2. $R$ contains no covered terms.
3. The terms of $R$ are in increasing order.



THEOREM 1: Every regular expression denoting a non-initial definite event has a unique canonical form.

PROOF: By definition, every regular expression denoting a definite event can be put in a definite form. This form can then be simplified by removing covered terms. If the definite terms are arranged in increasing order, we have a canonical form. To prove the uniqueness, suppose an expression $R$ has two canonical forms

$$R_1 = i^*(u_1 + u_2 + \ldots + u_p), \tag{9}$$

$$R_2 = i^*(v_1 + v_2 + \ldots + v_q). \tag{10}$$

Suppose further that a term $u_a$ of $R_1$ is not identical to any of the $v_j$ of $R_2$. Since $u_a \in R$, then $u_a \in R_2$; that is, $u_a$ must end in some $v_a$, or $u_a = w_a v_a$, for some $w_a \neq \lambda$. Now $v_a$ cannot be a term of $R_1$ because $v_a$ would cover $u_a$ in $R_1$. Since $v_a \in R_2$ then $v_a \in R_1$; that is, $v_a$ must end in some $u_b$, or $v_a = w_b u_b$, for some $w_b \neq \lambda$. This argument is repeated and, in each step, the length of the sequence which is a term of one of $R_1, R_2$ but not of the other is decreased by at least one. The argument stops when the sequence in question is of length 1 and is contained in one of $R_1, R_2$ but not in the other. This leads to the conclusion that $R_1 \neq R_2$, contradicting the assumption that $R_1$ and $R_2$ are both canonical forms for the same set of sequences. Therefore, for every $u_j$ in $R_1$ there is an identical $v_j$ in $R_2$ and vice versa. Thus the two forms are identical.

As an example, consider the expression $R_1 = i^*(01 + 100 + 101)$. The term 101 is redundant because $101 = 1(01)$; that is, it is covered by 01. The canonical form is therefore $R = i^*(01 + 100)$, and denotes all sequences ending in 01 or 100.

## V. DERIVATIVES OF REGULAR EXPRESSIONS

We shall introduce now the notion of derivative of a regular expression.[13] (This notion was introduced previously[16, 18, 19] in different terminology, but was not applied to regular expressions.) The use of derivatives (derived expressions) leads directly to the minimal state diagram.

DEFINITION 2: The derivative of a regular expression $R$, with respect to a sequence $a$, of unit length, $a \in A_k$, is denoted by $D_a[R]$ and is defined recursively by:

A) $D_a[a] = \lambda$; (11)

$D_a[b] = \phi$, for $b \in A_k$, $b \neq a$, or $b = \lambda$ or $b = \phi$. (12)

B) If $P$ and $Q$ are regular expressions, then

$D_a[P + Q] = D_a[P] + D_a[Q]$; (13)

$$D_a[PQ] = D_a[P]Q + \delta(P)\, D_a[Q]; \tag{14}$$

$$D_a[P^*] = D_a[P]P^*, \tag{15}$$

where $\delta(P) = \lambda$, if $\lambda \epsilon P$ and $\delta(P) = \phi$, if $\lambda \notin P$.

DEFINITION 3: The derivative, $D_s[R]$, of a regular expression R with respect to a sequence of input symbols, $s = a_1, a_2 \cdots a_r$ is defined by:

$$\begin{aligned} D_{a_1 a_2}[R] &= D_{a_2}\left[D_{a_1}[R]\right], \\ D_{a_1 a_2 a_3}[R] &= D_{a_3}\left[D_{a_1 a_2}[R]\right], \\ &\cdots\cdots\cdots\cdots\cdots\cdots \\ D_{a_1 \cdots a_r}[R] &= D_{a_r}\left[D_{a_1 \cdots a_{r-1}}[R]\right]. \end{aligned} \tag{16}$$

For completeness, we define

$$D_\lambda[R] = R. \tag{17}$$

The construction of derivatives will be illustrated by a simple example. Let $R = (0+1)^*1$. From Definition 2, we have:

$$\begin{aligned} D_a[R] &= D_a[(0+1)^*1], \\ &= D_a[(0+1)^*]\,1 + D_a[1], \quad \text{(since } \lambda \epsilon (0+1)^*\text{)}, \\ &= D_a[0+1]\,(0+1)^*1 + D_a[1], \\ &= (D_a[0] + D_a[1])\,(0+1)^*1 + D_a[1]. \end{aligned}$$

In particular, for $a = 0, 1$, the derivatives are:

$$D_0[R] = (\lambda + \phi)(0+1)^*1 + \phi,$$

$$D_1[R] = (\phi + \lambda)(0+1)^*1 + \lambda.$$

The above expressions can be simplified, using the identities (2) - (4), for $\phi$ and $\lambda$, with the result

$$D_0[R] = (0+1)^*1 = R,$$

$$D_1[R] = (0+1)^*1 + \lambda = R + \lambda.$$

The derivatives with respect to longer sequences can be found from Definition 3. For example,



$$\begin{aligned} D_{10}[R] &= D_0[D_1[R]] \\ &= D_0[R+\lambda] = D_o[R] + D_o[\lambda] \\ &= R + \phi = R. \end{aligned}$$

The derivatives and their properties are discussed in detail in Reference 13. We shall state some of the important characteristics of derivatives without proof.

Properties of Derivatives

1. The derivative $D_s[R]$ of a regular expression R with respect to any sequence s is itself a regular expression.

2. $t \epsilon D_s[R]$ if and only if $st \epsilon R$, where t is a sequence.

3. $s \epsilon R$ if and only if $\lambda \epsilon D_s[R]$.

4. Every regular expression R has only a finite number $d_R$ of distinct (i. e., not equivalent) derivatives.

5. If the derivatives are constructed for sequences of increasing length r beginning with $\lambda$, then all of the $d_R$ distinct derivatives are found for

$$r \leq d_R - 1.$$

6. Every regular expression can be written in the form:

$$R = \delta(R) + \sum_{a=0}^{k-1} aD_a[R], \tag{18}$$

where $\delta(R)$ is defined as in Definition 2 above. The term $aD_a[R]$ is the set of all sequences in R beginning with a.

7. Every regular expression R, with $d_R$ distinct derivatives, can be completely characterized by a set of $d_R$ equations of the form:

$$D_s[R] = \delta(D_s[R]) + \sum_{a=0}^{k-1} aD_{u_a}[R], \tag{19}$$

where $D_{u_a}$ is either $D_{sa}$, or a previously found derivative, equivalent to $D_{sa}$. The equations of the form (19) are called the characteristic equations of R.

8. An equation of the form

$$R = AR + B, \quad \lambda \notin A, \tag{20}$$

where A, B, R are regular expressions, has the unique (up to equiv-)

alence) solution:

$$R = A^*B. \tag{21}$$

9. The set of characteristic equations of R can be solved for R uniquely (up to equivalence), using property 8.

10. Two regular expressions are similar, if and only if one can be transformed to the other using only the identities

$$R + R = R,\ (P+Q) + R = P + (Q+R),\ P+Q = Q+P;$$

otherwise they are dissimilar. Every regular expression has only a finite number of dissimilar derivatives. This implies that the construction of derivatives will terminate, even if only similarity of derivatives is recognized.

In the construction of the derivative of a product RQ, it must be known whether R contains $\lambda$. This question can be answered if we know the length $L_m[R]$ of the shortest sequence contained in R, because $\lambda \epsilon R$ if and only if $L_m[R] = 0$. The length of the shortest sequence of R can be determined by inspection of R using the following rules:

$$\begin{aligned}
&L_m[\lambda] = 0,\\
&L_m[a] = 1,\ a \in A_k,\\
&L_m[\phi] = \infty,\\
&L_m[P+Q] = \min(L_m[P],\ L_m[Q]),\\
&L_m[P\cdot Q] = L_m[P] + L_m[Q],\\
&L_m[P^*] = 0.
\end{aligned}$$

A very useful interpretation of the derivative $D_s[R]$ is obtained by considering $D_s[R]$ as the set of all the sequences that can follow s in R. In other words, given s, $D_s[R]$ denotes the set of all sequences t, such that the sequence st is contained in R; that is, $D_s[R]$ denotes the set $T = \{t \mid st \in R\}$.

Derivatives of non-initial definite expressions have certain special properties. Let the general form of a definite expression be

$$R = i^* \sum_j f_j = i^*F. \tag{22}$$

Then the following result holds:



THEOREM 2: The derivative of a canonical non-initial definite expression R with respect to a sequence s either is $D_s[R] = R$, or else is of the form

$$D_s[R] = R + E_s, \tag{23}$$

where $E_s$ is an initial event and

$$R \,\&\, E_s = \phi. \tag{24}$$

(The symbol & denotes intersection: i. e., if P and Q are regular expressions denoting sets of sequences $S_P$ and $S_Q$, then P & Q denotes the set of sequences common to $S_P$ and $S_Q$.)

PROOF: Consider the derivative with respect to a sequence of length 1. We have

$$\begin{aligned}
D_a[R] &= D_a[i^*]F + D_a[F],\\
&= D_a[i]i^*F + D_a[F],\\
&= \lambda i^*F + D_a[F],\\
&= i^*F + D_a[F] = R + D_a[F].
\end{aligned} \tag{25}$$

Similarly, we find

$$\begin{aligned}
D_{ab}[R] &= D_b[R + D_a[F]],\\
&= R + D_b[F] + D_{ab}[F].
\end{aligned} \tag{26}$$

It is clear that $D_s[R]$ will always consist of R and several derivatives of F. The derivatives of F are clearly initial; i. e., we have the required form $R + E_s$. For example, $D_{ab}[R] = R + E_{ab}$ where $E_{ab} = D_b[F] + D_{ab}[F]$. In case $E_s = \phi$, we have $D_s[R] = R$.

It remains to be shown that $R \,\&\, E_s = \phi$. We assume $E_s \neq \phi$: otherwise the statement is trivially true. Let e be any sequence of $E_s$. Now, if $e \in R$, then there must be a sequence u such that $e = uf$, $f \in F$. In other words, if R contains a sequence e, then by the definition of a definite event, that sequence must end in some term of F. But $e \in E_s$, that is, e is a derivative of some sequence g in F. Thus $g = ve$, where v is of length at least 1, and $g \in F$. Hence we obtain $g = ve = vuf$, where both g and f are members of F. Clearly, this is a contradiction, since f covers g and R is assumed to be canonical. Thus no sequence of $E_s$ can be a member of R. This completes the proof of Theorem 2.

THEOREM 3: A regular expression R denotes a non-initial definite event if and only if R satisfies an equation of the form

$$R = iR + F, \tag{27}$$

where F is initial and $(iR) \,\&\, F = \phi$. Furthermore, $R = i^*F$ is the canonical form of R.

PROOF: If a regular expression R satisfies an equation of the form given in (27), then $R = i^*F$ by property 8 above, i.e., R is non-initial definite. Conversely, if R is non-initial definite, then it has a unique canonical form $R = i^*E$, which satisfies the equation $R = iR + E$. We must show that $(iR) \,\&\, E = \phi$. For all $a \in A_k$ we have $D_a[R] = R + D_a[E]$, where $R \,\&\, D_a[E] = \phi$, by Theorem 2. We assume here that $\lambda \notin R$, for otherwise $R = i^*$, and is of no interest. In view of property 6 we can write

$$\begin{aligned} R &= \sum_{a=0}^{k-1} aD_a[R], \\ &= \sum_{a=0}^{k-1} a(R + D_a[E]), \\ &= \Big(\sum_{a=0}^{k-1} a\Big) R + \sum_{a=0}^{k-1} aD_a[E]. \end{aligned} \tag{28}$$

The first term reduces to iR, and the second term is just the expansion (19) of E. In other words, we have $R = iR + E$. Since $R \,\&\, D_a[E] = \phi$, then $(aR) \,\&\, (aD_a[E]) = \phi$, and hence $(iR) \,\&\, E = \phi$. Thus we have proved that, if R is non-initial definite, it satisfies an equation of the form of eq. (27).

It remains to be shown that if $R = iR + F$ with $(iR) \,\&\, F = \phi$, then $R = i^*F$ is the canonical form of R. We must prove that no sequence in F covers any other sequence. Suppose $f_1, f_2 \in F$ and $f_1 = uf_2$, $u \neq \lambda$. Let $u = av$, $a \in A_k$. Since $f_2 \in R$ and R is non-initial definite, then $vf_2 \in R$. Therefore $avf_2 \in iR$; that is, $uf_2 = f_1 \in iR$. Thus iR and F both contain $f_1$, and this is a contradiction. This completes the proof of Theorem 3.

## VI. MINIMAL STATE DIAGRAMS FOR NON-INITIAL DEFINITE EVENTS

There exist algorithms[8, 10, 13] for constructing a state diagram directly from a regular expression; however, in general, these methods do not lead to a minimal state diagram. The method of derivatives,[13] for example, fails to yield a minimal state diagram, because the equivalence of two regular expressions may be difficult to recognize. In the case of non-initial definite events and certain other types of events (to be discussed later), these difficulties do not arise, if canonical expressions are used.



Suppose we wish to construct a state diagram for a regular expression R. The corresponding automaton M is in its starting state $q_\lambda$, before any sequences are applied. Let the sequences u and v take the automaton to states $q_u$ and $q_v$, respectively. Clearly, if the set of sequences which can be applied to M in state $q_u$ in order to produce an output, is identical to the set of sequences which can be applied to M in $q_v$, then $q_u$ and $q_v$ can be identified as a single state. In terms of derivatives, this means that if $D_u[R] = D_v[R]$ then $q_u$ and $q_v$ correspond to the same state. Suppose an input sequence s applied to an automaton M in state $q_u$ produces an output sequence $Z_u(s)$, and s applied to M in state $q_v$ produces an output sequence $Z_v(s)$. The states $q_u$ and $q_v$ are indistinguishable, if and only if $Z_u(s)$ and $Z_v(s)$ are identical for all s. It can be shown[8, 13] that the following theorem applies:

THEOREM 4: Two states $q_u$ and $q_v$ of an automaton M described by a regular expression R are indistinguishable if and only if $D_u[R] = D_v[R]$.

Using this result we can now describe an algorithm leading directly to the minimal state diagram from a canonical non-initial definite expression. (A similar theorem and construction algorithm were developed earlier in different terminology by Perles, Rabin and Shamir.[14])

### Construction of a Moore State Diagram

Given $R = i^*F$, $F \neq \lambda$, R canonical:

1. Introduce a starting state $q_\lambda$, with output $Z = 0$. This corresponds to the derivative $D_\lambda[R] = R$.

2. Find $D_0[R] = R + D_0[F]$. If $D_0[F] = \phi$, then $D_0[R] = D_\lambda[R]$. Introduce a transition under input 0 from $q_\lambda$ to $q_\lambda$. If $D_0[F] \neq \phi$, then $D_0[R] \neq D_\lambda[R]$ and a new state $q_0$ must be introduced. The output associated with $q_0$ is $Z = 1$ if and only if $\lambda \in D_0[R]$.

3. Find $D_1[R] = R + D_1[F]$. Compare $D_1[R]$ with $D_\lambda[R]$ and $D_0[R]$ (unless $D_0[R] = D_\lambda[R]$). If $D_1[R]$ is the same as $D_\lambda[R]$ (or $D_0[R]$), introduce a transition under $x = 1$ from $q_\lambda$ to $q_\lambda$ (or to $q_0$). If $D_1[R]$ is different from all the previous derivatives, introduce a new state $q_1$ and a transition under $x = 1$ from $q_\lambda$ to $q_1$. The output for $q_1$ is $Z = 1$, if and only if $\lambda \in D_1[R]$.

4. Repeat step 3 for the remaining sequences of length 1, that

is, for $a = 2, 3, \ldots, k-1$. A derivative $D_s[R]$ is called a characteristic derivative of $R$ if and only if there are no previously found derivatives identical to $D_s[R]$. Each time a new derivative is found it is compared with all the previous characteristic derivatives. Note that this comparison consists of a comparison of finite sets of sequences. This is true because every $D_s[R]$ is of the form $R + E_s$, where $E_s$ is finite and $R \,\&\, E_s = \phi$. Thus we need compare only the $E_s$ portion of $D_s[R]$.

5. If there are no characteristic derivatives with respect to sequences of length 1, the construction ends, and the minimal state diagram consists of one state. Otherwise, go to step 6.

6. If $D_a[R]$ is the first characteristic derivative (first among the derivatives with respect to sequences of length 1), find $D_{a0}[R]$, $D_{a1}[R], \ldots, D_{a(k-1)}[R]$. If $D_{ab}[R]$ is a characteristic derivative introduce state $q_{ab}$ and a transition under $x = b$ from $q_a$ to $q_{ab}$. If $D_{ab}[R] = D_s[R]$, previously found, introduce a transition from $q_a$ to $q_s$ under $x = b$.

7. Repeat step 6 for the remaining characteristic derivatives with respect to sequences of length 1.

8. Repeat steps 6 and 7 for derivatives with respect to sequences of length $r = 2, 3$, etc. The process terminates when for some $r$ there are no new characteristic derivatives (see properties 4 and 5 of Section V). The output associated with a state $q_s$ is $Z=1$, if and only if $\lambda \in D_s[R]$.

Example 1: $R = (0 + 1)^*01$, Moore Model

$D_\lambda[R] = R$, introduce $q_\lambda/0$.

$D_0[R] = R+1$, introduce $q_0/0$.

$D_1[R] = R = D_\lambda[R]$, return to $q_\lambda$.

$D_{00}[R] = R+1 = D_0[R]$, return to $q_0$.

$D_{01}[R] = R+\lambda$, introduce $q_{01}/1$ (since $\lambda \epsilon D_{01}[R]$)

$D_{010}[R] = R+1 = D_0[R]$, return to $q_0$.

$D_{011}[R] = R = D_\lambda[R]$, return to $q_\lambda$.

The construction terminates here because no new characteristic derivatives are found for sequences of length 3. The characteristic derivatives of $R$ are $D_\lambda$, $D_0$, $D_{01}$ corresponding to states $q_\lambda$, $q_0$, $q_{01}$. Figure 2 shows the minimal Moore state diagram. The heavy arrows and the associated expressions under the states represent the corresponding derivatives.



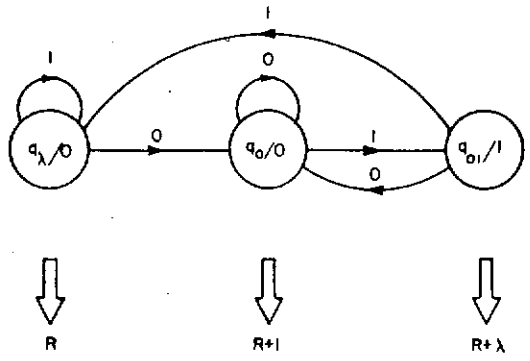


Fig. 2 Moore state diagram for $R = i^*01$

Construction of a Mealy State Diagram

The same method applies to a Mealy diagram with the exceptions:

1. The outputs are associated with transitions. A transition from $q_u$ to $q_v$ under $x = a$ (where $D_{ua}[R] = D_v[R]$), has output $z = 1$ if and only if $\lambda \epsilon D_{ua}[R]$.

2. If a derivative $D_u[R]$ differs from a previous characteristic derivative $D_v[R]$ by $\lambda$ only, then a new state need not be introduced for $D_u[R]$. This follows because, in the Mealy model, the presence of $\lambda$ in $D_u[R]$ indicates that the sequence $u$ produces an output, but tells us nothing about the future sequences that can follow $u$.

We shall use the same example to illustrate the construction of a Mealy state diagram.

Example 2: $R = (0 + 1)^*01$, Mealy Model.

$D_\lambda[R] = R$, $q_\lambda$.

$D_0[R] = R+1$, $q_0$.

$D_1[R] = R$, return to $q_\lambda$.

$D_{00}[R] = R+1$, return to $q_0$.

$D_{01}[R] = R+\lambda$, return to $q_\lambda$ with $z=1$.

The process terminates for the Mealy model, because no new states are introduced for sequences of length 2. The graph is minimal with 2 states, and is shown in Fig. 3. It should be noted that a non-initial event of length $q$ on $n$ inputs can be always realized using

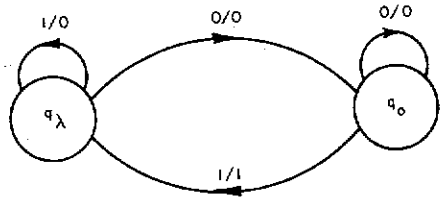


Fig.3 Mealy state diagram for R = i*01

n(q - 1) delay elements, i.e., by remembering each input for the past q - 1 units of time. For example, R = i*(02 + 012 + 112) can be realized with 2 × 2 = 4 delay elements. The minimal state graph (Mealy) for R has 3 states, hence requires only 2 delay elements.

## VII. COMPOSITE DEFINITE EVENTS

If a definite event is initial, it is clear that the form R = F cannot be simplified (except possibly by factoring) provided that no two sequences of F are identical. An initial definite expression is canonical if it is a finite sum of distinct finite sequences arranged in increasing order. Obviously such a form is unique.

We shall now consider the most general type of definite event, namely the composite event which is neither initial nor non-initial. A regular expression is said to be in a (composite) definite form, if it is in the form,

$$R = E + i^*F = \sum_{j=1}^{m} e_j + i^* \sum_{k=1}^{n} f_k \,. \tag{29}$$

An expression of the form (29) can be reduced by using the following two rules:

Rule 1: If a term $e_j$ (or $f_k$) is covered by a non-initial term $f_r$, i.e., $e_j = sf_r$ ($f_k = sf_r$), then $e_j$ ($f_k$) should be removed.

Rule 2: If, for a given initial term $e_j$, $R \supset i^*e_j$, then $e_j$ becomes a non-initial term and all other terms ending in $e_j$ are removed.

We shall illustrate the reduction rules by an example. Consider

$$R = 0 + 102 + 212 + i^*(00 + 02 + 10 + 20 + 002),$$

over the alphabet $\{0, 1, 2\}$. The application of Rule 1 shows that the terms 102 and 002 are covered by the non-initial term 02. The removal of these terms yields



$$R = 0 + 212 + i^*(00 + 02 + 10 + 20).$$

This expression can be simplified further by Rule 2, because

$$R \supset 0 + i^*(0 + 1 + 2)0 = 0 + i^*10 = (\lambda + i^*i)0 = i^*0.$$

Thus the initial term 0 becomes non-initial and all the other terms ending in a 0 are removed, with the result:

$$R = 212 + i^*(0 + 02).$$

In general, to determine whether Rule 2 can be used, each initial term $e_j$ is tested to check whether $R \supset i^*e_j$. This check can be done by inspection, as illustrated above.

DEFINITION 4: A regular expression R denoting a definite event is in a canonical definite form if and only if

1. R is in definite form, $R = E + i^*F$.
2. R cannot be simplified by Rules 1 and 2.
3. The terms of E and F are arranged in increasing order.

THEOREM 5: Every regular expression denoting a definite event has a unique canonical form, $R = E + i^*F$, $E \,\&\, (i^*F) = \phi$, irreducible by Rules 1 and 2.

PROOF: Assume R is irreducible by Rule 1 and that $s \in E$ and $s \in i^*F$. This implies that $s = uf$, for some $f \in F$. But f covers $s \in E$; hence R is reducible by Rule 1, contrary to the assumption. Hence $E \,\&\, (i^*F) = \phi$.

We shall now prove the uniqueness. Suppose $P = E + i^*F$ and $Q = G + i^*H$ are two equivalent canonical forms. Assume that $e \in E$ but $e \notin G$. Then e must end in some term of H, $e = sh$, for $e \in P = Q$. Now $H \supset i^*h$; therefore $H \supset i^*sh = i^*e$. Thus we have the contradiction that $e \in E$ and $P \supset i^*e$, contrary to Rule 2. Therefore $E \subset G$, and by the same argument, $G \subset E$, or $E = G$. This implies that $P = Q$ if and only if $i^*F = i^*H$, for we have $E \,\&\, (i^*F) = \phi$. Now $i^*F$ and $i^*H$ are irreducible by Rule 1; hence $i^*F$ and $i^*H$ are in non-initial canonical form, which has been proved unique. Thus $P = Q$ if and only if $E = G$ and $F = H$, that is, the forms are identical.

It is clear that the derivative of a composite definite event in canonical form is of the form

$$D_s[R] = K_s + i^*F. \tag{30}$$

where $K_s$ is initial; i.e., the derivative of a definite event is a definite event.

THEOREM 6: If $R = E + i^*F$ cannot be simplified by Rule 1, then $D_S[R] = K_S + i^*F$ cannot be simplified by Rule 1, and hence $K_S \,\&\, (i^*F) = \phi$.

PROOF: Consider the derivative with respect to a sequence $a \in A_k$ of length 1. Then we have

$$D_a[R] = D_a[E] + D_a[F] + i^*F$$

$$= D_a[E+F] + i^*F$$

Suppose $k \in D_a[E+F] = K_a$ and $k = uf$, $f \in F$. Then $ak \in E+F$, and $ak = auf$ is covered by $f$. This is a contradiction; hence $D_a[R]$ is irreducible by Rule 1, and $K_a \,\&\, (i^*F) = \phi$, as in Theorem 5. Since $D_{ab}[R] = D_b[D_a[R]]$, the proof is extended by induction to derivatives with respect to sequences of any length.

It is not true, in general, that the derivative of a canonical definite expression is canonical, for Rule 2 may be violated. For example, if $R = i^*(001 + 101)$, then $D_0[R] = 01 + i^*(001 + 101)$ can be reduced by Rule 2 to $D_0[R] = i^*01$.

For the purposes of constructing minimal state graphs using derivatives, we must ensure that the equivalence of any two derivatives is recognized. This can always be done, provided that the given expression is irreducible by Rule 1 (Rule 2 is unnecessary). Thus, given $R = E+i^*F$, irreducible by Rule 1, the derivatives should be found in the form $D_S[R] = K_S + i^*F$, and Rule 2 should not be applied to any derivatives. Then, if $D_t[R] = K_t + i^*F$, $D_S[R] = D_t[R]$, if and only if $K_S = K_t$, in view of Theorem 6. This last test on $K_S$ and $K_t$ involves the comparison of finite sets of finite sequences and presents no difficulties.

The following example illustrates the construction of the minimal Mealy state diagram for $R = 10 + i^*01$. We have

$$D_\lambda = 10 + i^*01, \quad q_\lambda.$$

$$D_0 = 1 + i^*01, \quad q_0.$$

$$D_1 = 0 + i^*01, \quad q_1.$$

$$D_{00} = 1 + i^*01, \quad q_0.$$

$$D_{01} = \lambda + i^*01, \quad q_{01}.$$

$$D_{10} = \lambda + 1 + i^*01, \quad q_0.$$



$$D_{010} = 1 + i^*01, \quad q_0.$$

$$D_{011} = i^*01, \quad q_{01}.$$

The resulting graph is shown in Fig. 4, and is minimal with 4 states. The original expression could have been written in different

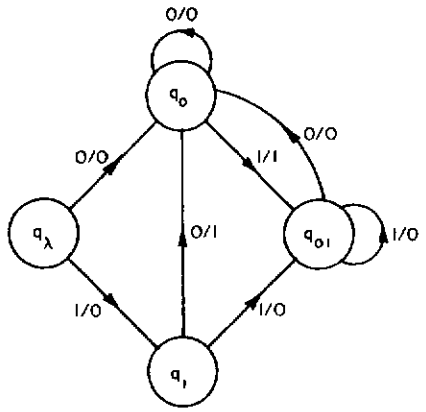


Fig. 4 State graph for $R = 10 + i^*01$

definite form, irreducible by Rule 1. For example,

$$R = 01 + 10 + i^*(001 + 101).$$

The reader can convince himself that the graph constructed from this expression is identical to that of Fig. 4. The first few derivatives are

$$D_\lambda = 01 + 10 + i^*(001 + 101), \quad q_\lambda,$$

$$D_0 = 1 + 01 + i^*(001 + 101), \quad q_0,$$

$$D_1 = 0 + 01 + i^*(001 + 101), \quad q_1,$$

etc.

## VIII. REVERSE DEFINITE EVENTS

We shall now consider a class of events, related to definite events, for which simple canonical forms can be found.

DEFINITION 5: The reverse of a regular expression R is denoted by $R^-$ and is defined recursively as follows:

$$x^- = x, \quad \text{for } x \in A_k \text{ or } x = \phi \text{ or } x = \lambda, \tag{31}$$

$$(P + Q)^- = P^- + Q^-, \tag{32}$$

$$(PQ)^- = Q^-P^-, \tag{33}$$

$$(P^*)^- = (P^-)^*. \tag{34}$$

It can be verified easily that the reverse of a regular expression R has the properties:

1. $R^-$ is a regular expression.
2. $s \in R$, if and only if $s^- \in R^-$.
3. $(R^-)^- = R$.

DEFINITION 6: An event R is reverse definite if and only if $R^-$ is definite, i.e., if and only if R can be written in the form

$$R = E + Fi^*,$$

where E and F are initial.

In other words, a reverse definite event consists of an initial event E and of all the sequences beginning with a specified finite set F of finite sequences. It is clear that reverse definite events can be simplified by rules analogous to Rules 1 and 2 (Section VII). We have:

Rule $1^-$: If $g \in E+F$, $f \in F$, and $g = fu$, then f covers g and g can be removed.

Rule $2^-$: If $e \in E$ and $R \supset ei^*$, then e becomes a member of F and all other terms of E+F beginning with e can be removed.

Using arguments analogous to those of Theorem 5, we can prove the following:

THEOREM 7: Every reverse definite event has a unique canonical form $R = E+Fi^*$, $E \,\&\, (Fi^*) = \phi$, not reducible by Rules $1^-$ and $2^-$.

The derivatives of reverse definite events are reverse definite. In this case we can prove a stronger theorem about derivatives than Theorem 6 for definite events.

THEOREM 8: The derivative $D_s[R]$ of any canonical reverse definite regular expression is a canonical reverse definite expression (provided the sequences are ordered).



PROOF: Consider first $D_a[R]$, $a \in A_k$. Let $R = E+Fi^*$ be canonical. If $R = i^*$ then the canonical form is $\phi + \lambda i^*$ and

$$D_a[R] = \phi + \lambda i^*,$$

that is, $D_a[R]$ is canonical. Hence assume $R \neq i^*$, which is equivalent to saying that $\lambda \notin F$. Then

$$D_a[R] = D_a[E] + D_a[F]i^*.$$

Suppose $u \in D_a[E] + D_a[F]$ and $u = vp$, $v \in D_a[F]$; then $au \in E+F$, that is, $avp \in E + F$ and $av \in F$. Thus R is reducible by Rule $1^-$, which is a contradiction. Hence the derivative is irreducible by Rule $1^-$. Now let $u \in D_a[E]$ and suppose that $D_a[R] \supset ui^*$. Then $au \in E$ and $R \supset aui^*$, contrary to the assumption that R is irreducible by Rule $2^-$. Thus the process of taking derivatives preserves both rules; that is, $D_a[R]$ is canonical if the sequences of $D_a[E]$ and $D_a[F]$ are ordered. By induction $D_s[R]$ is canonical if R is, for s of any length.

As a consequence of Theorem 8, the state graph corresponding to any reverse definite event can be constructed directly from the canonical form by the derivative method. State graphs for reverse definite events have certain special properties which we proceed to describe.

We shall consider Moore state diagrams in the following discussion. A state $q_j$ of a reduced state diagram is terminal if and only if all the transitions out of $q_j$ return to $q_j$; that is, once state $q_j$ is entered, the automaton remains in $q_j$. A graph is called a cascade graph[12, 15] if and only if it has no feedback loops; i.e., starting at any node it is impossible to return to that node. We shall consider the empty graph to be cascade.

DEFINITION 7: Given a reduced state graph G of an automaton M, we define $G_e$ as the explicit graph of M, where $G_e$ is obtained from G by the following rules:

1. If G has a terminal state $q_\phi$ with output $Z = 0$, remove $q_\phi$ and all the transitions leading to $q_\phi$.
2. If G has a terminal state $q_I$ with output $Z = 1$, remove all the transitions from $q_I$ back to $q_I$; that is, remove all the self-loops on $q_I$.
3. The remaining states and transitions of G constitute $G_e$.

It is clear that the explicit graph $G_e$ is uniquely constructed from G. Since we have assumed that G is reduced, there can be at most one terminal state $q_\phi$ and at most one terminal state $q_I$. This follows because all transitions from $q_\phi(q_I)$ lead to $q_\phi(q_I)$ since

$D_s[\phi] = \phi$, ($D_s[i^*] = i^*$) for all $s$, where the derivative associated with $q_\phi(q_I)$ is clearly $\phi(i^*)$.

THEOREM 9: An event is reverse definite if and only if the explicit graph obtained from its reduced state diagram is cascade.

PROOF: 1) First assume that an explicit graph $G_e$ of a reduced graph $G$ is cascade. If there are no states of $G_e$ with output 1, then $G$ represents the reverse definite event $R = \phi + \phi i^*$. Next let us assume that $G_e$ has states $q_1, q_2, \ldots, q_r$ with output $Z = 1$ where no $q_j$ is terminal. For each $q_j$ there is only a finite number of sequences $e_{ij}$ leading from the initial state $q_\lambda$ to $q_j$. Let

$$E = \sum_{i,j} e_{ij} .$$

Then $E$ describes precisely the set of all sequences which produce an output when applied to the automaton represented by $G$ and leave $G$ in one of the $q_j$. This follows because $G_e$ is cascade and it is impossible to return to $q_j$. If $G_e$ has no other states with $Z = 1$, then $E$ is a regular expression representing $G$. Since $E$ consists of a finite set of finite sequences, $E$ is reverse definite with the canonical form $R = E + \phi i^*$. Now suppose that $G_e$ has a state $q_I$ with $Z = 1$, where $q_I$ was a terminal state of $G$. Let the set of sequences leading from $q_\lambda$ to $q_I$ in $G_e$ be

$$F = \sum_k f_k .$$

This set is clearly finite, for $G_e$ is assumed to be cascade. Then the regular expression $R = E + Fi^*$ denotes the set of all sequences which produce an output in $G$. Hence $G$ corresponds to a reverse definite event.

2) It remains to be shown that, if $R$ is a reverse definite event, then the explicit graph obtained from the reduced state diagram of $R$ is cascade. By Theorem 8, $R$ has a unique canonical form, $R = E + Fi^*$. If $R = \phi$, then its explicit graph is empty. If $R = i^*$, that is, $\lambda \in F$, $G_e$ consists of state $q_I$ with no transitions, hence is trivially cascade. Assume now that $\lambda \notin F$ and that $E + F$ is not empty. Suppose that $G_e$ for $R$ has a feedback loop. This implies the existence of sequences $u$ and $uv$ both leading from $q_\lambda$ to some state $q_u$. This in turn implies that $D_u[R] = D_{uv}[R]$. From this last relation we see that

$$D_{uvv}[R] = D_v[D_{uv}[R]] = D_v[D_u[R]] = D_{uv}[R] = D_u[R] .$$

In general, $D_{uv^n}[R] = D_u[R]$. Choose $n$ large enough so that the length of $uv^n$ exceeds the length of the longest sequence in $E + F$.



$D_u[R]$ cannot be empty, for $D_u[R] = \phi$ implies $q_u = q_\phi$, and hence $q_u$ cannot appear in $G_e$. Let $x \in D_u[R]$; then $ux \in R$ and $uv^n x \in R$. Now $uv^n x$ is too long to be a member of $E$; hence $uv^n$ must begin with some $f \in F$, for $n$ large enough, that is, $uv^n = fp$, for some $p$. Let $u = u_1 u_2$, $v = v_1 v_2$ be any two partitions of $u$ and $v$, where one of $u_1, u_2$ ($v_1, v_2$) can be $\lambda$. The first possibility is $u_1 = f$. But, if $u_1 = f$, then $D_{u_1}[R] = i^*$, because $f \in F$. Hence the sequence $u_1 u_2 v$ cannot appear in $G_e$ because $q_u = q_I$ and all transitions from $q_I$ have been removed. The second possibility is $uv^r v_1 = f$, $r \leq n$. Here $D_{uv^r v_1}[R] = D_{uv_1}[R] = i^*$. Hence the sequence $uv_1 v_2$ cannot appear in $G_e$, unless $v_2 = \lambda$, for $uv_1$ takes $G_e$ to $q_I$. This leaves $f = uv^{r+1}$ implying that $D_{uv^{r+1}}[R] = D_u[R] = i^*$. Again $uv$ cannot appear as a sequence of $G_e$ for $q_u = q_I$. Hence $G_e$ contains no loops, i.e., $G_e$ is cascade.

As a consequence of Theorem 9, it is exceedingly easy to obtain the canonical form for a reverse definite event. This follows as a result of the following theorem.

THEOREM 10: The regular expression

$$R = \sum_{i,j} e_{ij} + (\sum_k f_k)i^*$$

obtained in Part 1 of the proof of Theorem 9 is the canonical regular expression for $R$, provided the sequences are ordered.

PROOF: Let $R = E + Fi^*$ be the expression derived from $G_e$ Suppose $e \in E$, $f \in F$ and $e = fp$. Since $D_f[R] = i^*$, this is impossible, for $fp$ cannot be a sequence of $G_e$. Suppose $f_j, f_k \in F$, $f_j \neq f_k$ and $f_j = f_k p$. This is impossible, by the same argument. Hence $R$ cannot be reduced by Rule 1⁻. Now suppose $e \in E$ and $R \supset ei^*$. Then $e$ leaves $G_e$ in $q_I$ and hence $e$ cannot be a member of $E$. This follows from the construction in the proof of Theorem 9. Thus $R$ is irreducible by Rule 2⁻. Therefore $R$ is canonical, provided the terms of $E$ and $F$ are ordered.

We conclude the discussion of reverse definite events by presenting an example. Let $R = 1 + (01 + 10)i^*$. The state diagram is shown on Fig. 5(a); Fig. 5(b) is the explicit graph for $R$, showing clearly the canonical form of $R$.

## IX. TESTING FOR DEFINITENESS; CONVERSION TO CANONICAL FORMS

When a given definite expression arises from a word description of a problem, it is likely to appear in definite form. For example, the designer may require an output for all sequences ending in a member

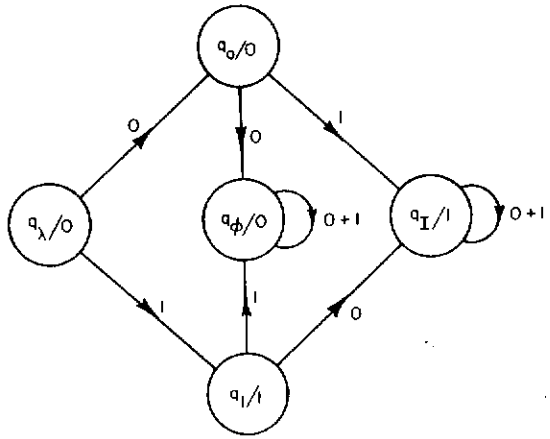


Fig. 5 State diagrams for $R = 1 + (01 + 10)i^*$. (a) State diagram; (b) Explicit diagram.

of a specified set of sequences. On the other hand, the expression may appear in a different form and it may be desirable to convert it to a simple canonical form.

In certain cases, it may be possible to use algebraic identities to accomplish the conversion. For example, consider

$$R = (1 + 00^*1)00^*1 .$$

If R is non-initial definite, then it must be possible to convert it to the form $R = i^*F$. Since F cannot contain any starred terms, a logical first step is to use the identity $R^*R = RR^*$ with the result $R = (1 + 00^*1)^*0^*01$. Now R is of the form $R = P01$,



$$P = (1 + 00^*1)^*0^* ,$$

and P contains $\lambda$. If we can prove that $P = (0 + 1)^*$, then R is definite. This can be done as a result of the following theorem.

THEOREM 11: A regular expression R is equivalent to $i^* = I$, if and only if every derivative of R contains $\lambda$.

PROOF: If $\lambda \in D_s[R]$ for all s, then $s \in R$ (see Property 3 of Section V). Thus R contains all sequences, that is, $R = i^*$. On the other hand, if $\lambda \notin D_s[R]$ for some s, then $s \notin R$ and $R \neq i^*$.

Since every regular expression has only a finite number of distinct derivatives, the theorem implies an effective procedure. In the above example,

$$D_\lambda[P] = P = (1+00^*1)^*0^* , \quad \lambda \in D_\lambda[P].$$
$$D_0[P] = 0^*1P + 0^* , \quad \lambda \in D_0[P] .$$
$$D_1[P] = P , \quad \lambda \in D_1[P].$$
$$D_{00}[P] = 0^*1P + 0^* = D_0[P] , \quad \lambda \in D_{00}[P].$$
$$D_{01}[P] = P, \quad \lambda \in D_{01}[P] .$$

Because there are no new derivatives for sequences of length 2, we have found all the characteristic derivatives and each contains $\lambda$. Hence $R = i^*01$.

There are cases where such a simple approach may not work. For example, in the expression

$$R = (2+12+(0+10+11)(0+1)^*2)^*(0+10+11)(0+1)^*2 ,$$

it is not clear which algebraic transformations are to be used. In general, it is not known what constitutes a complete set of identities and this subject requires further investigation. In order to handle all cases, we present a method based on the state diagram of regular expressions.

In the preceding section we have shown that the test for determining whether a reduced state diagram corresponds to a reverse definite event is trivial: the explicit graph must be cascade. We can apply this result to testing for definiteness, if we can construct a

graph for $R^-$ given the graph for R. We proceed now to describe the solution to this problem.

Consider a (Moore) state diagram G with states $q_\lambda, q_2, \ldots q_m$. The state $q_\lambda$ is the starting state and all the other states are accessible from $q_\lambda$, by assumption. Let the set of states of G with output Z = 1 be

$$\pi_\lambda = \{p_1, p_2, \ldots, p_r\}.$$

The set of all sequences leading from $q_\lambda$ to some $p_j \in \pi_\lambda$ is denoted by a regular expression R. We wish to construct a state diagram $G^-$ representing the sequences of $R^-$. We shall first describe the construction* and then prove its validity.

The graph $G^-$ will have states $\pi_s$ corresponding to sets of states of G. The starting state of G is

$$\pi_\lambda = \{p_1, p_2, \ldots, p_r\},$$

the set of output states of G. An input $a \in A_k$ applied to $\pi_\lambda$ takes $G^-$ to $\pi_a(\pi_\lambda)$, the "a-predecessor of $\pi_\lambda$." Given a set of states $\pi_s$ of G we define $\pi_t(\pi_s)$, the t-predecessor of $\pi_s$, as the set of states of G, $\pi_t(\pi_s) = \{q_x\}$ such that $q_x \in \pi_t$ if and only if the sequence t applied to $q_x$ in graph G takes G to $q_y \in \pi_s$. It is clear that any t-predecessor of $\pi_\lambda$ written $\pi_t(\pi_\lambda)$ or simply $\pi_t$ consists of all those states $q_y$ of G, such that t applied to $q_y$ takes G to an output state. It follows also that $\pi_t(\pi_s) = \pi_{ts}$. Thus the proposed reverse graph $G^-$ will consist of $\pi_\lambda$ and of all its predecessors. If $\pi_s$ is empty, it corresponds to the empty set of states of G, $\pi_s = \phi$. If $\pi_{as}$ is the a-predecessor of $\pi_s$, then in the graph $G^-$ the input $a \in A_k$ takes $\pi_s$ to $\pi_{as}$. An output Z = 1 is associated with $\pi_s$ of $G^-$, if and only if $q_\lambda \in \pi_s$.

An example will be used to illustrate the construction. In Fig. 6(a), state $q_3$ is the only output state; hence $\pi_\lambda = \{q_3\}$. Since $q_3$ has no 0-predecessors, $\pi_0 = \phi$. Similarly,

$$\pi_1 = \{q_2\},\ \pi_{01} = \pi_0(\pi_1) = \{q_2\},\ \pi_{11} = \{q_\lambda, q_3\},\ \text{etc.}$$

The only set containing $q_\lambda$ is $\pi_{11} = \{q_\lambda, q_3\}$; hence this is the only output state of $G^-$.

In the following theorem it is proved that the graph constructed according to the above description corresponds to $R^-$.

* This construction can be viewed as a special case of the reduction of nondeterministic automata to deterministic automata, as discussed by Rabin and Scott.[16]



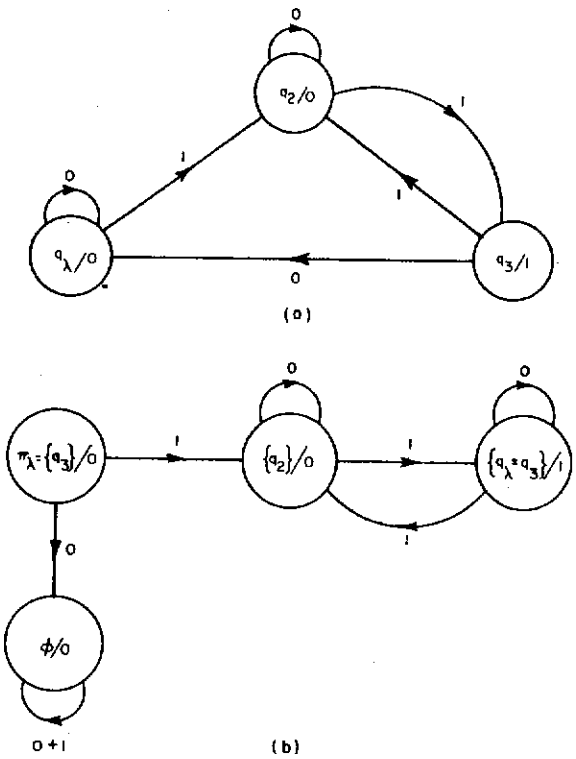


Fig. 6 Construction of reverse graph. (a) Graph G; (b) Reverse graph $G^-$.

THEOREM 12*: Given a graph G representing a regular expression R, the graph $G^-$, in which the predecessors of $\pi_\lambda$ correspond to states, represents the regular expression $R^-$, the reverse of R. *Note: It is not necessary for G to be a reduced graph.*

PROOF: Let $s \in R$; then s takes G from $q_\lambda$ to an output state $p_j$. Hence, by construction, $q_\lambda \in \pi_s$. Let P denote the regular expression describing $G^-$. Then for every $s \in R$, $s^- \in P$. Conversely, if $s^- \in P$, then $s^-$ takes $G^-$ from $\pi_\lambda$ to $\pi_s$. But $q_\lambda \in \pi_s$, because the output associated with $\pi_s$ is Z = 1, for $s^- \in P$. Hence the sequence s applied to $q_\lambda$ in G must lead to an output state, because $q_\lambda$ is in the s-predecessor of $\pi_\lambda$. Hence for every $s^- \in R^-$, $s \in P$. Therefore $P = R^-$.

* This theorem, in different terminology, was also proved by Rabin and Scott.[16]

We shall now prove a rather surprising result about the construction algorithm for $G^-$:

THEOREM 13: Given any (not necessarily reduced) state diagram G (with starting state $q_\lambda$ and all other states accessible from $q_\lambda$), the graph $G^-$ of predecessors of $\pi_\lambda$ is always reduced, if only one state is introduced for each predecessor of $\pi_\lambda$. In other words, two states of $G^-$ are equivalent if and only if their predecessor sets are identical.

PROOF: Suppose states $\pi_s$ and $\pi_t$ of $G^-$ are equivalent. This implies $D_{s^-}[R^-] = D_{t^-}[R^-]$. If both derivatives are empty then the predecessor sets must also be empty, that is, $\pi_s = \pi_t = \phi$. Now suppose $D_{s^-}[R^-]$ is not empty; then there is at least one state

$$q_u \epsilon \pi_s \ .$$

If u takes G from $q_\lambda$ to $q_u$, then $us \epsilon R$. But this implies that $s^-u^- \epsilon R^-$. Now, since we have assumed equivalence of $D_{s^-}[R^-]$ and $D_{t^-}[R^-]$, we have also $t^-u^- \epsilon R^-$, $ut \epsilon R$. But this implies that $q_u \epsilon \pi_t$. Hence $\pi_t \supset \pi_s$, and by a similar argument, $\pi_s \supset \pi_t$. Therefore $\pi_s = \pi_t$. Thus we have shown that, if two states are equivalent their predecessor sets are identical. The converse is true by construction, i. e., two identical predecessor sets are automatically represented by a single state in $G^-$.

In view of the above results, the test for definiteness of G is reduced to a test for reverse definiteness of $G^-$. Thus a graph G represents a definite event if and only if the explicit reverse graph is cascade. Since the reverse graph is always reduced, Theorem 10 applies. This means that the canonical form for $R^-$ can be obtained by inspection from $G^-$. If $R^- = E + Fi^*$, then $R = E^- + i^*F^-$ gives the canonical form for R.

We conclude with an example. Fig. 7(a) shows a graph for

$$R = (2 + 12 + (0 + 10 + 11)(0 + 1)^*2)^*(0 + 10 + 11)(0 + 1)^*2 \ .$$

The graph for $R^-$ is given in Fig. 7(b). It is seen that R is non-initial definite and that the canonical form for R is

$$R = (0 + 1 + 2)^*(02 + 012 + 112) \ .$$

Other methods for testing for definiteness are described by Perles, Rabin and Shamir, [14] and by McCluskey. [17]

## X. THE DEGREE OF DEFINITE EVENTS

Perles, Rabin and Shamir[14] have defined the degree of a definite event. For completeness we shall show the meaning of the degree in terms of canonical forms. First, however, we shall give their definition of a definite event. Let L(s) denote the length of the sequence s. The k-suffix of s ($L(s) \geq k$) is defined to be the sequence consisting of the last k symbols of s.

DEFINITION 8: Let $k \geq 0$, and let R be a regular expression. Then R is weakly k-definite, if for any s with $L(s) \geq k$, $s \epsilon R$, if and only if the k-suffix of s is in R.



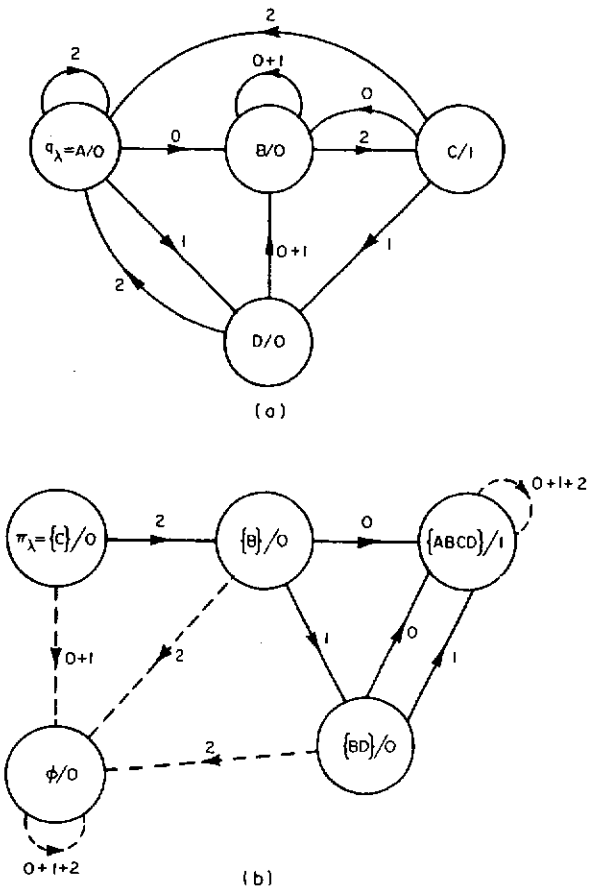


Fig. 7 Illustrating the example. (a) State diagram; (b) Reverse state diagram.

DEFINITION 9: R is k-definite, if it is weakly k-definite, but not weakly (k - 1)-definite.

DEFINITION 10: If R is k-definite, its degree is denoted by deg R and defined to be k, i.e., deg R = k.

The degree of a definite expression in canonical form can be found by inspection, as a result of the following:

THEOREM 14: Let $R = E + i^*F$ be canonical, i.e., irreducible by rules 1 and 2 of Section VII. Then $k = \max(L_M[E] + 1, L_M[F])$, is the degree of R, where $L_M[X]$ is the length of the longest sequence in X, and $L_M[X] = 0$, if X is empty.

PROOF: Consider any sequence s, $L(s) \geq k$, and let v be the k-suffix of $s = uv$. If $s \in R$, then clearly $s \notin E$, since $L_M[E] < k$. Thus s must be of the form $s = pf$, $f \in F$, where $L(f) \leq k$. Hence v must be of the form $v = qf$, for $L(v) = k$. But $R \supset i^*f \supset qf = v$. Therefore, if $s \in R$, $L(s) \geq k$, its k-suffix is also in R. Now suppose $s \notin R$ but $v \in R$. Since $L(v) = k$, $v \in R$ implies that $v = tf$, for some t and some $f \in F$. Thus $s = uv = utf \in R$, which is a contradiction. Hence R is weakly k-definite.

Suppose now that R is weakly (k - 1)-definite. Let $f_i \in F$, $L(f_i) > k - 1$, and let v be the (k - 1)-suffix of $f_i$. Then $v \in R$ and $R \supset i^*v$. It is clear that $v \notin E$, for rule 2 would be violated. Thus we must have $v = pf_j$, $f_j \in F$. This, however, violates rule 1, because $f_j$ covers $f_i$. Hence, there are no sequences $f_i \in F$ with $L(f_i) > k - 1$, if R is to be weakly (k - 1)-definite, i.e., $L_M[F] < k$. Consider now a sequence $e_j \in E$, with $L(e_j) \geq k - 1$, and let v be the (k - 1)-suffix of $e_j$. If R is to be weakly (k - 1)-definite, then $R \supset i^*v \supset i^*e_j$. This violates rule 2, implying that $L_M[E] + 1 < k$. Thus we have arrived at a contradiction, that $L_M[E] + 1 < k$, $L_M[F] < k$, proving that R is not weakly (k - 1)-definite. Therefore R is k-definite.

It is interesting to note that the degree of a definite event is not the same, in general, as the length of a definite event, which we have defined as $\max(L_M[E], L_M[F])$. The length of a definite event R in canonical form corresponds to the minimum length of the past input sequence that must be inspected, in order to determine whether R occurs or not.

Perles, Rabin and Shamir[14] also express definite events in the form $R = E + i^*F$, but with the restriction that $L_M[E] < k$ and $L(f_j) = k$, for all $f_j \in F$. For example, our expression $R = 111 + i^*0$, in their form would be



$$R = 111 + 0 + 00 + 10 + 000 + 010 + 100 + 110 + i^*(0000 + 0010 + 0100 + 0110 + 1000 + 1010 + 1100 + 1110).$$

## XI. OPERATIONS ON DEFINITE EVENTS

It can be verified that the class of definite events is closed under union (+), intersection (&), and complementation ('). In general, it is not easy to convert a regular expression with (&) and ('), to a form which does not contain these operators.[10] This is not a difficult problem for definite events in canonical form. For example,

$$(11 + i^*0)' = ((11 + 0 + 00 + 10) + i^*i^20)'$$

$$= \lambda + 1 + 01 + i^*i^21 = \lambda + 1 + i^*(01 + 011 + 111).$$

It is clear that the class of definite events cannot be closed under both concatenation (•) and the star operation (*). If it were, then all regular events would be definite, and this is not the case. We shall investigate whether the class of definite events is closed under either one of these operations. †

In general, the class of definite events is not closed under the star operation; e.g., $(01)^*$ is not definite. Similarly, the class is not closed under concatenation; e.g., $i^*1i^*1$ is not definite. However, it is interesting to note that the star of any non-initial definite event is definite, for we have:

$$(i^*E)^* = \lambda + i^*E(i^*E)^*$$

$$= \lambda + i^*E + (i^*E)^*i^*E.$$

Since $i^*E \supset (i^*E)^*i^*E$, we have

$$(i^*E)^* = \lambda + i^*E.$$

Thus starring a non-initial event simply adds $\lambda$ to it. If we consider the definite iterate operation[6, 11]

$$R^+ = RR^*,$$

then

$$(i^*F)^+ = i^*F(i^*F)^* = i^*F + (i^*F)^*i^*F = i^*F.$$

Thus the class of non-initial definite events is closed under the defi-

† This problem has been suggested by S. Ginsburg.

nite iterate operation.

Similarly, there are cases where the product of two definite events is definite. For example, let $P = \lambda + E + i^*i^n$, $Q = G + i^*H$, where E, G and H are finite sets. Then

$$PQ = G + EG + i^*i^nG + (\lambda + E)i^*H + i^*i^ni^*H$$

$$= G + EG + i^*i^nG + i^*H + i^*i^nH$$

$$= G + EG + i^*(H + i^nG),$$

which is definite. These are, however, rather special cases.

## XII. CONCLUDING REMARKS

We have considered a restricted class of regular expressions corresponding to definite events. Because of this restriction it was possible to show the existence of simple canonical forms and to construct minimal state diagrams directly from these forms. The investigation of reverse definite events has lead to a simple procedure for testing for definiteness.

The solutions to the above problems in the general case are not yet known.

## ACKNOWLEDGMENT

The author wishes to thank Professor E. J. McCluskey, E. B. Eichelberger and J. F. Poage of Princeton University, Princeton, N. J., for the many useful comments and discussions of the subject.

## REFERENCES

1. E.F. Moore, "Gedanken-Experiments on Sequential Machines," in *Automata Studies*, C.E. Shannon and J. McCarthy, eds. (Princeton, N.J.: Princeton University Press, 1956), Study 34, pp. 129-153.
2. G.H. Mealy, "A Method for Synthesizing Sequential Circuits," *Bell Syst. Tech. J.*, Vol. 34, No. 5, p. 1045 (September 1955).
3. E.J. McCluskey, Jr., "Logical Design of Digital Circuits," Class notes for E417, Department of Electrical Engineering, Princeton University, Princeton, N.J.
4. E.J. McCluskey, Jr., "A Comparison of Sequential and Iterative Circuits," *Trans. AIEE, Pt. 1--Comm. and Electronics*, Vol. 78, pp. 1039-1044 (January 1960).
5. D.A. Huffman, "The Synthesis of Sequential Switching Circuits," *J. Franklin Inst.*, Vol. 257, Nos. 3 and 4, pps. 161 and 275 (March and April 1954).
6. S.C. Kleene, "Representation of Events in Nerve Nets and Finite Automata," in *Automata Studies*, C.E. Shannon and J. McCarthy, eds. (Princeton, N.J.: Princeton University Press, 1954), Study 34, pp. 3-41.



7. I.M. Copi, C.L. Elgot, and J.B. Wright, "Realization of Events by Logical Nets," *J. Assoc. Comp. Mach.*, Vol. 5, pp. 181-196 (April 1958).
8. C.Y. Lee, "Automata and Finite Automata," *Bell Syst. Tech. J.*, Vol. 39, p. 1267 (September 1960).
9. D. N. Arden, "Delayed Logic and Finite State Machines," *Proc. 2nd Ann. Symp. on Switching Circuit Theory and Logical Design*, Detroit, Mich., pp. 133-151 (October 1961). Also in *Theory of Computing Machine Design*, (Ann Arbor, Mich.: University of Michigan Press, 1960), pp. 1-35.
10. R. McNaughton and H. Yamada, "Regular Expressions and State Graphs for Automata," *IRE Trans. on Electronic Computers*, Vol. EC-9, pp. 39-47 (March 1960).
11. J. A. Brzozowski, "A Survey of Regular Expressions and their Applications," *IRE Trans. on Electronic Computers*, Vol. EC-11, pp. 324-335 (June 1962).
12. J. A. Brzozowski and E. J. McCluskey, Jr., "Signal Flow Graph Techniques for Sequential Circuit State Diagrams," *IEEE Trans. on Electronic Computers*, Vol. EC-12 (April 1963).
13. J.A. Brzozowski, "Properties of Regular Expressions and State Diagrams," Tech. Report No. 15, Department of Electrical Engineering, Digital Systems Laboratory, Princeton University, Princeton, N.J., March 1962.
14. M. Perles, M.O. Rabin, and E. Shamir, "The Theory of Definite Automata," Tech. Report No. 6, The Hebrew University of Jerusalem, Applied Logic Branch, August 1961.
15. S.J. Mason, "Feedback Theory--Properties of Signal Flow Graphs," *Proc. IRE*, Vol. 41, No. 9, pp. 1144-1156 (September 1953).
16. M. O. Rabin and D. Scott, "Finite Automata and their Decision Problems," *IBM J. Res. and Dev.*, Vol. 3, pp. 114-125 (April 1959).
17. E. J. McCluskey, "Reduction of Feedback Loops in Sequential Circuits and Carry Leads in Iterative Networks," *Proc. 3rd Ann. Symp. on Switching Theory and Logical Design*, Chicago, Ill., pp. 91-102 (October 1962). Also Tech. Rept. 28, Dept. of Elec. Engrg., Digital Systems Lab., Princeton University, Princeton, N. J., (August 1962).
18. G. N. Raney, "Sequential Functions," *J. Assoc. Comp. Mach.*, Vol. 5, p. 177 (April 1958).
19. C. C. Elgot and J. D. Rutledge, "Operations on Finite Automata," *Proc. 2nd Ann. Symp. on Switching Theory and Logical Design*, Detroit, Mich. (October 1961).