INFORMATION AND CONTROL 31, 75–96 (1976)

Feedback and Generalized Logic

E. S. BAINBRIDGE

Department of Mathematics, University of Ottawa, Ottawa, Ontario, Canada K1N 6N5

Although the distinction between software and hardware is a posteriori, there is an a priori distinction that masquerades as the software–hardware distinction. This is the distinction between procedure interconnection, the semantics of flow chart diagrams, which is known to be described by the regular expression calculus; and system interconnection, the semantics of network diagrams, which is described by a certain logical calculus, dual to a calculus of regular expressions. This paper presents a proof of the duality in a special case, and gives the interpretation of the logical calculus for sequential machine interconnection. A minimal realization theorem for feedback systems is proved, which specializes to known open loop minimal realization theorems.

INTRODUCTION

The distinction in computer science between *hardware* and *software* has been assumed to be based on the engineering choice between the realization of an algorithm by a special purpose or a general purpose device. In other words, the hardware–software distinction is generally regarded as an a posteriori dichotomy. Nevertheless, there remains a persistent feeling that the distinction is of conceptual and not simply engineering significance. This paper proposes that there is an a priori distinction that masquerades as the a posteriori hardware–software distinction, and is of fundamental conceptual importance. Briefly, it is the following. Hardware systems are typically described by black box diagrams (henceforth called *network* diagrams), whereas software systems are typically described by *flowchart* diagrams. There is a syntactic duality and a semantic duality between networks and flowcharts. Moreover, there is a logical calculus for the semantics of network interconnection which is dual to a calculus of regular expressions for the semantics of flowchart interconnections. This result in its full generality will appear elsewhere; this paper presents a special case of the general duality, and the specialization of the logical interconnection calculus

to the case of sequential machines. It is observed that since linear systems interconnect according to the regular expression calculus, they lie (in a suitably linearized setting) on the *flowchart* side of the duality. Linearized *networks* are *multilinear* systems. It is my suspicion that the real difference between linear and nonlinear systems is that they fall on opposite sides of this duality; the proper distinction may be between systems which interconnect according to the regular expression calculus or the logical calculus.

Sections 1 through 5 and Section 8, which deal with the points mentioned so far, do not assume any category theory. Section 6 describes Lawvere's observations on generalized logic, and necessarily presumes category theoretical concepts. It must be understood that the formulation of Section 5 is not an ad hoc construction, nor was it discovered by generalizing the formulation of Section 3. Rather, it must be seen in the context of the generalized logical calculus of Section 6.

Section 7 uses some of the adjunctions of Section 6 to describe a minimal realization theory for feedback connections of sequential machines which specializes to known open loop minimal realization theories.

1. Flowcharts and Networks

Figure 1a is a typical schematic *flowchart*, that is, an interconnection of *procedures*. Specifically, it is an example of a schematic *iteration loop*. (We are assuming in this section that these terms already have some intuitive content for the reader; formal definitions will not be given here.) Figure 1b is a typical schematic *network*, that is, an interconnection of *systems*. Specifically, it is a schematic *feedback loop*. This pair of diagrams illustrates a syntactic duality between networks and flowcharts. A diagram is dualized by reversing arrows (and, to conform to pictorial conventions, reflected in a diagonal). Thus, exit is dual to input, entry is dual to output, junction of

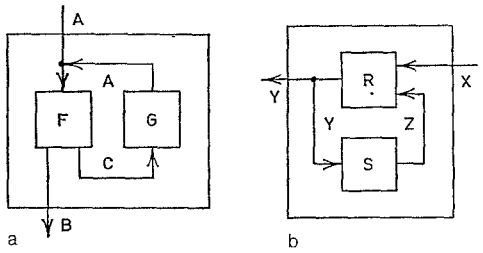

Fig. 1. a. Flowchart. b. Network.

control paths is dual to bifurcation of wires, iteration is dual to feedback, and so on.

To make this somewhat clearer, observe that one could define a purely syntactic flowchart diagram language which would have the junction of paths (Fig. 2a) along with some collection of labeled multiple entry and exit boxes as atomic formulae, and would have alternation (Fig. 3a) as a binary operation and iteration (Fig. 4a) as a unary operation. Note that the sequencing operation can be obtained by combining these primitive operations (Fig. 5a); indeed, any flowchart can be obtained by appropriate iteration operations on the alternation of its components.

Dually, one could define a syntactic network diagram language which would have the bifurcating wire (Fig. 2b) along with labeled multiple input

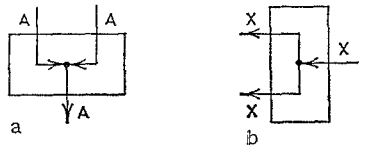

FIG. 2. a. Junction of paths. b. Bifurcating wire.

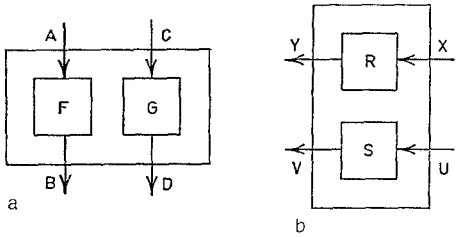

FIG. 3. a. Alternate procedures. b. Parallel systems.

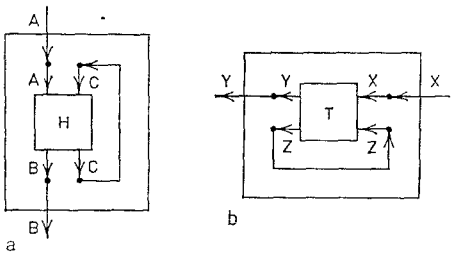

FIG. 4. a. Iteration. b. Feedback.

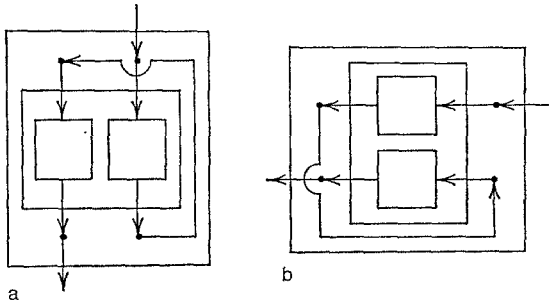

FIG. 5. a. Procedures in sequence. b. Systems in series.

and output boxes as atomic formulae, and would have parallel connection (Fig. 3b) as a binary operation and feedback connection (Fig. 4b) as a unary operation. Figure 5b shows the construction of a serial connection; in general, any network can be obtained by suitable feedback operations on the parallel connection of its components. These languages are syntactically dual in an obvious sense. The accompanying semantic duality can now be outlined.

We shall give in Section 2 a translation of the flowchart language into a language of regular expressions. We understand that the syntactic manipulation rules for regular expressions are part of this interpretation of the flowchart language in the sense that the only semantics for the flowchart language which we permit shall be those in which the rules are valid. A particular example of such a further interpretation is given in Section 2, namely the standard semantics of flowcharts described by Eilenberg (1974). On the other hand, we shall give in Section 3 a translation of the network language into a certain language of logical formulae. We understand that certain syntactic deduction rules for these formulae are part of this interpretation of the network language in the sense that we shall permit only those semantics for the network language in which the deduction rules are valid. A particular example of such a further interpretation is given in Section 5, namely the formulation of sequential machine interconnection using generalized logic.

The most general statement of the semantic duality between flowcharts and networks concerns the abstract regular expression semantics and the abstract logical formula semantics. Various particular duality principles are obtained by taking suitable pairs of further interpretations. In this introductory paper we shall discuss only such a particular duality principle, but one in which the important features of the general situation appear. The regular expression calculus is interpreted as in Section 2, and the logical calculus is given a corresponding interpretation in Section 3. There is an

interpretation of the regular expression calculus which corresponds to the interpretation of the logical calculus in Section 5, and this gives another duality principle with an interesting meaning. This will be discussed in a later paper.

2. Semantics of Flowchart Interconnection

This interpretation of flowcharts is a paraphrasing of Eilenberg's (1974) notion of a machine. In Fig. 1a, let $F \subset A \times (B + C)$ be a binary relation from A to $B + C$ (disjoint union) and let $G \subset C \times A$ be a binary relation from C to A. F and G may be interpreted as the relations (for example, partial functions) computed by certain procedures. With initial data $a \in A$, box F computes a result $b \in B$, or a result $c \in C$, or no result at all. If the result is in C, box G computes a result $a' \in A$, or no result at all. This process is repeated and terminates if a result in B is eventually produced. If $\circ$ denotes ordinary relational composition and * denotes transitive reflexive closure (i.e., for $H \subset D \times D$, $H^* = I_D \cup H \cup H \circ H \cup H \circ H \circ H \cup \cdots$, where I_D is the identity relation on D), then we may express the relation computed by the flowchart as follows. Since $A \times (B + C) \cong A \times B + A \times C$ we may write $F = F_B{}^A \cup F_C{}^A$, where $F_B{}^A = F \cap A \times B$ and $F_C{}^A = F \cap A \times C$. Figure 6 shows the steps the computation may take, with their associated relations. Thus one sees that the relation computed is $(F_C{}^A \circ G)^* \circ F_B{}^A$, a regular expression in $F_C{}^A$, G, $F_B{}^A$.

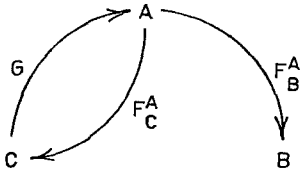

FIG. 6. Computation paths of Fig. 1.

More systematically, consider the following interpretation of the abstract flowchart language. An atomic formula (procedure) P with entries labeled $A_1, \ldots, A_m$, and exits labeled $B_1, \ldots, B_n$ will be represented by an $m \times n$ matrix $(P_j{}^i)$. If there is no ambiguity, we may write $P_j{}^i$ as $P_D{}^C$ when $A_i = C$, $B_j = D$. The junction of paths box is the matrix $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ where 1 is the regular expression denoting the empty word. Assuming inductively that formulae F, G, H of the language have been interpreted as matrices over the semiring

of regular expressions in the components P_j^i of atomic formula, we define the alternation $F + G$ (Fig. 3a) of F and G by

$$F + G = \begin{bmatrix} F & 0 \\ 0 & G \end{bmatrix}$$

where 0 is a suitable matrix in the expression for the empty set. In general, if H has an entry and an exit with the same label, C say, there is an iteration operation which identifies them. For simplicity, suppose H has entry labels A, C and exit labels B, C. The iteration operation $\circ^C$ (Fig. 4a) is defined by

$$\circ^C H = H_B{}^A \vee H_C{}^A (H_C{}^C)^* H_B{}^C.$$

In the general case, $H_C{}^A$ would become a column, $H_B{}^C$ a row; $\vee$ and concatenation become matrix sum and product.

The usual flowchart semantics is consistent with this interpretation of the language. A formula with entry labels $A_1, \ldots, A_m$, and exit labels $B_1, \ldots, B_n$ represents a relation from $A_1 + \cdots + A_m$ to $B_1 + \cdots + B_n$ where $+$ denotes disjoint union. Such a relation is uniquely specified by its component relations from A_i to B_j. Thus, given an interpretation of the atomic formulae as relations, and interpreting the junctions of paths as the graph ∇_A of the codiagonal function $A + A \to A$, it is easy to check that alternation and iteration are computed according to the above formulas. In particular, if Fig. 1a is expressed as a formula in the language, one obtains the expression $(F_C{}^A \circ G)^* \circ F_B{}^A$ by the above formalism, modulo well-known regular expression identities.

3. Memoryless Interpretation of Networks

The interpretation of Fig. 1a as an interconnection of sequential machines will be given in Section 5. Here, to introduce the logical interconnection calculus and the special case of the duality principle, we consider a simpler interpretation. Let R and S in Fig. 1b be interpreted as relations (for example, partial functions) computed by some devices with negligible internal delay. With inputs $x \in X$ and $z \in Z$, box R either produces instantaneously some $y \in Y$, or else produces no output at all; and similarly for box S. Thus, an input $x \in X$ to the network produces an output $y \in Y$ if and only if there is some $z \in Z$ which satisfies the simultaneous conditions imposed by boxes R and S on x, y, z. For technical reasons we consider output–input relations rather than input–output relations; that is, we regard R as a subset of

$Y \times (X \times Z)$ and S as a subset of $Z \times Y$. If we use predicate notation for R and S, the output–input predicate of the network may be written as

$$\exists z\, R(y, x, z) \wedge S(z, y)$$

Thus there is a certain logical calculus for interconnection of these memoryless networks, and the usual semantics of these formulas gives the intended interpretation.

More systematically, we can interpret the formulas of the abstract network language as logical formulas, as follows. An atomic formula (system) Q with inputs labeled $X_1, \ldots, X_m$ and outputs labeled $Y_1, \ldots, Y_n$ will be represented by an $(n + m)$ary predicate $Q(y_1, \ldots, y_n, x_1, \ldots, x_m)$. The bifurcating wire is represented by the predicate $(y_1 = x) \wedge (y_2 = x)$. Assuming inductively that formulae R, S, T of the language have been interpreted as predicates, we define the parallel connection $R \wedge S$ (Fig. 3b) of R and S by

$$R \wedge S(y, v, x, u) = R(y, x) \wedge S(v, u)$$

where x, y, u, v represent lists of variables if necessary. In general, if T has an input and an output with the same label, Z say, there is a feedback operation which identifies them. Consider $T = T(y, z, x, z')$, where x, y may be lists of variables. The feedback operation $\exists Z$ (Fig. 4b) is defined by

$$\exists Z\, T(y, x) = \exists z\, T(y, z, x, z).$$

The semantics of memoryless network diagrams is consistent with this formalism, via the usual semantics of quantifier logic. The bifurcating wire has output–input relation $\Delta_X = \{((x, x), x) \mid x \in X\}$, and the operations $\wedge$ and $\exists Z$ acquire their proper meanings for memoryless network diagrams. In particular, if Fig. 1b is expressed as a formula in the language, one obtains the logical formula $\exists z\, R(y, x, z) \wedge S(z, y)$ by the above formalism, modulo well-known syntactic equivalences. In Section 5 we will see that another interpretation of this logical formalism gives the semantics of sequential machine interconnection.

4. Duality

To any $F \subset A \times B$ assign the relation $F^{\#} \subset 2^A \times 2^B$ defined by

$$F^{\#}(\alpha, \beta) = \forall a\, \forall b\, F(a, b) \Rightarrow (\alpha a \Rightarrow \beta b)$$

(where αa, βb are the truth values of the unary predicates $\alpha \in 2^A$, $\beta \in 2^B$

at $a \in A$, $b \in B$); and to any $\phi \subset 2^A \times 2^B$ assign the relation $\phi_\# \subset A \times B$ defined by

$$\phi_\#(a, b) = \forall\alpha\, \forall\beta\, \phi(\alpha, \beta) \Rightarrow (\alpha a \Rightarrow \beta b)$$

These functions constitute a *Galois connection* between the partially ordered sets (in fact Boolean algebras) $2^{A\times B}$ and $2^{2^A\times 2^B}$; that is,

$$\begin{aligned} F \subset G \quad &\text{implies} \quad G^\# \subset F^\#, \\ \phi \subset \psi \quad &\text{implies} \quad \psi_\# \subset \phi_\#, \\ F \subset \phi_\# \quad &\text{iff} \quad \phi \subset F^\#, \\ F^\#{}_\#{}^\# = F^\# \quad &\text{and} \quad \phi_\#{}^\#{}_\# = \phi_\#. \end{aligned}$$

This Galois connection arises in a standard way (Cohn, 1965) from the function

$$((a, b), (\alpha, \beta)) \mapsto (\alpha a \Rightarrow \beta b)\colon (A \times B) \times (2^A \times 2^B) \to 2.$$

Relative to this Galois connection, every $F \subset A \times B$ is *closed*, i.e., $F^\#{}_\# = F$. To show this, note first that for any $a \in A$, $F^\#(\alpha_a, \beta_a)$ holds, where α_a is $\_ = a$ and β_a is $F(a, \_)$. Thus, if $F^\#{}_\#(a, b)$ holds, then $\alpha_a(a) \Rightarrow \beta_a(b)$, so $F(a, b)$ holds. Thus, $F^\#{}_\# \subset F$; and we always have $F \subset F^\#{}_\#$, so $F^\#{}_\# = F$.

Thus, by general properties of Galois connections, $2^{A\times B}$ is anti-isomorphic to the subalgebra $\{\phi \subset 2^A \times 2^B \mid \phi_\#{}^\# = \phi\}$ of $2^{2^A\times 2^B}$.

DUALITY THEOREM. $(\nabla_A)^\# = \Delta_{2^A}$, *and* $F + G = (F^\# \wedge G^\#)_\#$, $\circ^C H = (\exists 2^C H^\#)_\#$.

Proof. Let $u_1, u_2\colon A \to A + A$ be the natural injections. Then $\nabla_A = \{(u_1 a, a) \mid a \in A\} \cup \{(u_2 a, a) \mid a \in A\}$. Now $(\nabla_A)^\# \subset 2^{A+A} \times 2^A \cong (2^A \times 2^A) \times 2^A$, so modulo this isomorphism we see that $((\alpha, \beta), \gamma) \in (\nabla_A)^\#$ if and only if $\alpha a = \gamma a$ and $\beta a = \gamma a$ for all $a \in A$; that is, $\alpha = \gamma = \beta$. Thus $(\nabla_A)^\# = \Delta_{2^A}$.

For the conjugacy of $+$ and $\wedge$, $\circ^C$ and $\exists 2^C$, since all subsets of $A \times B$ are closed, it is sufficient to prove that $(F + G)^\# = F^\# \wedge G^\#$ and $(\circ^C H)^\# = \exists 2^C H^\#$.

Note that $(F + G)^\# \subset 2^{A+C} \times 2^{B+D} \cong 2^A \times 2^C \times 2^B \times 2^D$. Then $(F + G)^\#(\alpha, \gamma, \beta, \delta)$ means that for all $x \in A + C$ and $y \in B + D$

$$\begin{aligned} &((x \in A \wedge y \in B \wedge F(x, y)) \vee (x \in C \wedge y \in D \wedge G(x, y))) \\ &\quad \Rightarrow [((x \in A \wedge \alpha x) \vee (x \in C \wedge \gamma x)) \Rightarrow ((y \in B \wedge \beta y) \vee (y \in D \wedge \delta y))] \end{aligned}$$

This is equivalent to

$$((x \in A \wedge y \in B \wedge F(x, y)) \Rightarrow (\alpha x \Rightarrow \beta y)) \wedge ((x \in C \wedge y \in D \wedge G(x, y)) \Rightarrow (\gamma x \Rightarrow \delta y)),$$

that is, to $F^{\#}(\alpha, \beta) \wedge G^{\#}(\gamma, \delta) = F^{\#} \wedge G^{\#}(\alpha, \gamma, \beta, \delta)$.

Rather surprisingly, the second equation is the principle of partial correctness proofs for programs. On the one hand, $(\circ^C H)^{\#}(\alpha, \beta)$ asserts that for all data $a \in A$, if αa holds and the flowchart of Fig. 4a produces a result $b \in B$ (i.e., $\circ^C H(a, b)$), then βb holds. On the other hand, $(\exists 2^C H^{\#})(\alpha, \beta) = \exists \gamma H^{\#}(\gamma, \alpha, \gamma, \beta)$ (recalling that since $H \subset (C + A) \times (C + B)$ we may interpret $H^{\#}$ as a subset of $2^C \times 2^A \times 2^C \times 2^B$). This asserts that there exists $\gamma \in 2^C$ such that for all $x \in C + A$, $y \in C + B$,

$$\begin{aligned}&(x \in C \wedge y \in C \wedge H(x, y) \Rightarrow (\gamma x \Rightarrow \gamma y))\\ &\quad \wedge (x \in C \wedge y \in B \wedge H(x, y) \Rightarrow (\gamma x \Rightarrow \beta y))\\ &\quad \wedge (x \in A \wedge y \in C \wedge H(x, y) \Rightarrow (\alpha x \Rightarrow \gamma y))\\ &\quad \wedge (x \in A \wedge y \in B \wedge H(x, y) \Rightarrow (\alpha x \Rightarrow \beta y)).\end{aligned}$$

Stated informally, if H produces a result from data satisfying γ or α, the result satisfies γ or β. It is a simple induction argument, familiar from program correctness proofs that this ensures that if the flowchart of Fig. 4a produces a result from data satisfying α, the result satisfies β. Thus, $\exists 2^C H^{\#} \subset (\circ^C H)^{\#}$.

For the converse, suppose $(\circ^C H)^{\#}(\alpha, \beta)$. Consider

$$\gamma c = (\exists a\, \alpha a \wedge H_C{}^A \circ (H_C{}^C)^*(a, c)) \wedge (\forall b\, (H_C{}^C)^* \circ H_B{}^C(c, b) \Rightarrow \beta b).$$

Case (i). Suppose $x \in C$ and $y \in C$ and $H(x, y)$ and γx. Then there is $a \in A$ such that αa and $H_C{}^A \circ (H_C{}^C)^*(a, x)$. Since $H_C{}^C(x, y)$, then $\exists a\, \alpha a \wedge H_C{}^A \circ (H_C{}^C)^*(a, y)$ holds. Let $b \in B$ be such that $(H_C{}^C)^* \circ H_B{}^C(y, b)$. Since $H_C{}^C(x, y)$, then $(H_C{}^C)^* \circ H_B{}^C(x, b)$ holds, so βb holds. Thus, $\forall b\, (H_C{}^C)^* \circ H_B{}^C(y, b) \Rightarrow \beta b$ holds. Thus, γy holds.

Case (ii). Suppose $x \in C$ and $y \in B$ and $H(x, y)$ and γx. Then there is $a \in A$ such that αa and $H_C{}^A \circ (H_C{}^C)^*(a, x)$. Since $H_B{}^C(x, y)$, then $H_C{}^A \circ (H_C{}^C)^* \circ H_B{}^C(a, y)$, so $(\circ^C H)(a, y)$. But αa holds, so by hypothesis, βy holds.

Case (iii). Suppose $x \in A$ and $y \in C$ and $H(x, y)$ and αx. Then $\exists a\, \alpha a \wedge H_C{}^A \circ (H_C{}^C)^*(a, y)$ holds, since $H_C{}^A(x, y)$ holds. Let $b \in B$ be such

that $(H_C{}^C)^* \circ H_B{}^C(y, b)$. Then $H_C{}^A \circ (H_C{}^C)^* \circ H_B{}^C(x, b)$ holds, so $(\circ^C H)(x, b)$ holds. Since αx holds, then by hypothesis, βb holds. Thus γy holds.

Case (iv). Suppose $x \in A$ and $y \in B$ and $H(x, y)$ and αa. Then $H_B{}^A(x, y)$, so $(\circ^C H)(x, y)$ holds. Since αx holds, then by hypothesis βy holds.

Thus $\exists\gamma\, H^{\#}(\gamma, \alpha, \gamma, \beta)$ holds so $(\circ^C H)^{\#} \subset \exists 2^C H^{\#}$. Therefore, the equations hold. Q.E.D.

We have exhibited a duality between the Eilenberg semantics of the regular expression calculus for flowchart interconnection and the semantics of the logical calculus for memoryless network interconnection (that is, the usual semantics for this $\exists$, $\wedge$ calculus).

5. Feedback for Sequential Machines

To every multiple input and output sequential machine can be assigned a suitable *biaction* of the output monoids on the left and the input monoids on the right. This construction is the crucial first step in developing a calculus of interconnection for sequential machines, a calculus that is syntactically identical to the logical calculus of Section 3. For simplicity of exposition, we consider only sequential machines of the form specified below, but the extension to the general case is entirely obvious.

A *multiple input and output sequential machine* with *inputs* X, Z and *outputs* Y, Z consists of a set Q of *states*, and functions

$$\delta: Q \times (X \times Z) \to Q,$$

$$\lambda: Q \times (X \times Z) \to (Y \times Z).$$

The *free monoid* generated by a set A is denoted A^* and has as its elements finite sequences of elements of A, with concatenation as multiplication and the empty sequence ϵ as identity.

A *configuration* of the above sequential machine is an element of $Y^* \times Z^* \times Q \times X^* \times Z^*$.

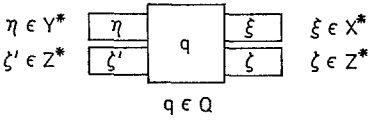

Fig. 7. A configuration.

We define an equivalence relation $\sim$ on configurations whereby each configuration is identified with its successor (if any) under the operation of the machine. Formally, the relation is generated by the identifications

$$(\eta, \zeta', q, x\xi, z\zeta) \sim (\eta\lambda_Y(q, x, z), \zeta'\lambda_Z(q, x, z), \delta(q, x, z), \xi, \zeta)$$

where $x \in X$, $z \in Z$, and λ_Y, λ_Z are the components of λ. Let $C = Y^* \times Z^* \times Q \times X^* \times Z^*$ be the set of configurations, and let $\bar{C} = C/\sim$ be the set of equivalence classes $[\eta, \zeta', q, \xi, \zeta]$ of configurations $(\eta, \zeta', q, \xi, \zeta) \in C$. Then $\bar{C}$ admits a left $Y^* \times Z^*$, right $X^* \times Z^*$ action, namely

$$(\eta_1, \zeta_1') \cdot [\eta, \zeta', q, \xi, \zeta] \cdot (\xi_1, \zeta_1) = [\eta_1\eta, \zeta_1'\zeta', q, \xi\xi_1, \eta\eta_1].$$

To show this, note that C has such an action, with an analogous definition. This action is compatible with the identifications generating the relation $\sim$, so by general principles of universal algebra $\sim$ is a congruence, and the action on $\bar{C}$ is well defined.

This action will be called the *characteristic biaction* of the machine. We can recover the functions δ, λ from the characteristic biaction, by noting that the equivalence classes may be identified with configurations having at least one input tape empty, and considering $[\epsilon, \epsilon, q, \epsilon, \epsilon] \cdot (x, z)$ for $x \in X$, $z \in Z$, $q \in Q$.

In order to see the analogy with Section 3, we must interpret the characteristic biaction as a *functor* $\Phi: (Y^* \times Z^*)^{\mathrm{op}} \times (X^* \times Z^*) \to \mathrm{Sets}$, analogous to the characteristic function of a relation $T: (Y \times Z) \times (X \times Z) \to 2$. We shall not, however, assume any category theory here; rather, we shall give a direct definition of the operation *coend* which is analogous to existential quantification.

Let M, N, P be monoids, and let Γ be a left $M \times N$, right $P \times N$ action on a set K. Then the *N-coend* of Γ, denoted $\int^N \Gamma$ is a left M, right P action on $K/\equiv$, where $\equiv$ is the equivalence relation generated by the identifications $k \cdot (1, n) \equiv (1, n) \cdot k$, $n \in N$, $k \in K$; which is defined by $m \cdot \bar{k} \cdot p = \overline{(m, 1) \cdot k \cdot (p, 1)}$, where $\bar{}$ denotes equivalence class relative to $\equiv$. One can see that this is well defined by noting that it is compatible with the identifications generating $\equiv$, since

$$\begin{aligned}(m, 1) \cdot (k \cdot (1, n)) \cdot (p, 1) &= (m, 1) \cdot k \cdot (p, n) \\ &= ((m, 1) \cdot k \cdot (p, 1)) \cdot (1, n) \equiv (1, n) \cdot ((m, 1) \cdot k \cdot (p, 1)) \\ &= (m, n) \cdot k \cdot (p, 1) = (m, 1) \cdot ((1, n) \cdot k) \cdot (p, 1).\end{aligned}$$

In this case, let us compute $\int^{*}\Phi$; that is, $\bar{C}/\equiv$ with the appropriate biaction. The equivalence $\equiv$ is generated by the identifications

$$[\eta, \zeta', q, \xi, \zeta] \cdot (\epsilon, \zeta'') \equiv (\epsilon, \zeta'')[\eta, \zeta', q, \xi, \zeta],$$

that is,

$$[\eta, \zeta', q, \xi, \zeta\zeta''] \equiv [\eta, \zeta''\zeta', q, \xi, \zeta]$$

Equivalently, $\bar{C}/\equiv = C/\approx$ where $\approx$ is the equivalence relation generated by the identifications generating $\sim$, together with additional identifications

$$(\eta, \zeta', q, \xi, \zeta\zeta'') \approx (\eta, \zeta''\zeta', q, \xi, \zeta) \tag{1}$$

These latter identifications are shown in Fig. 8. That is, the left end of the Z^* output tape is identified with the right end of the Z^* input tape—feedback! The left action of Y^* and right action of X^* are induced by the congruence $\approx$.

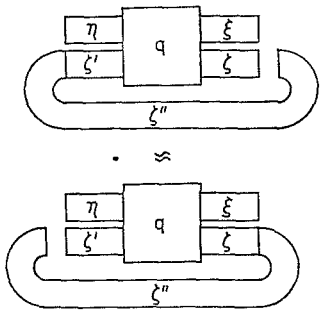

FIG. 8. The identifications (1).

Let us compare $\int^{Z^*}\Phi$ with the characteristic biaction of what would usually be taken as the feedback connection of the given machine. The Z output would be fed back with unit delay, giving the sequential machine

$$\bar{\delta}: (Q \times Z) \times X \to (Q \times Z), \qquad \bar{\lambda}: (Q \times Z) \times X \to Y$$

where $\bar{\delta}((q, z), x) = (\delta(q, x, z), \lambda_Z(q, x, z))$ and $\bar{\lambda}((q, z), x) = \lambda_Y(q, x, z)$. The characteristic biaction of $\bar{\delta}$, $\bar{\lambda}$ is a subaction of $\int^{Z^*}\Phi$. We forego a formal proof in favor of the more suggestive pictorial sequence of Fig. 9. In fact, the biaction $\int^{Z^*}\Phi$ contains a subaction isomorphic to the characteristic biaction of the machine δ, λ with feedback delay n, for every n; and $\int^{Z^*}\Phi$ is precisely the disjoint union of these subactions. In view of this, and the universal properties possessed by $\int^{Z^*}$ which are sketched in the next section,

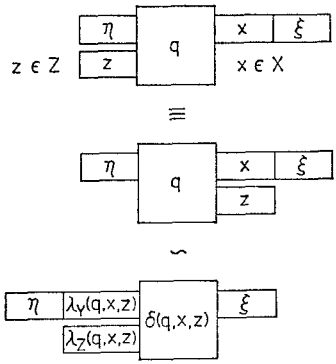

FIG. 9. 1-Delay feedback.

it would seem that we should consider not a single feedback machine associated with δ, λ, but rather the family of feedback machines that are captured in $\int^{Z^*} \Phi$.

The fact that interconnection operations are more appropriately viewed as operations on biactions than on the machines themselves is reiterated when one considers parallel interconnection. How should Fig. 3b be interpreted? Must one supply inputs to both machines simultaneously, or can they operate independently? If they have characteristic biactions Φ: $Y^{*\mathrm{op}} \times X^* \to$ Sets and Ψ: $V^{*\mathrm{op}} \times U^* \to$ Sets, the natural construction on biactions $\Phi \times \Psi$: $(Y^* \times V^*)^{\mathrm{op}} \times (X^* \times U^*) \to$ Sets gives the more general answer; that is, if Φ is an action on K and Ψ an action on L, the action $\Phi \times \Psi$ on $K \times L$ is given by

$$(\eta, \nu) \cdot (k, l) \cdot (\xi, \mu) = (\eta \cdot k \cdot \xi, \nu \cdot l \cdot \mu),$$

which allows action of simultaneous or independent inputs.

Thus, the logical interconnection calculus with the interpretation of coend and product yields the semantics of sequential machine interconnection (strictly, it gives a semantics that subsumes the usual semantics).

6. Generalized Logic

To explain the assertion that the operations of Section 5 participate in a generalized logical calculus, we state here the analogies with ordinary logic. The fact that set-valued functors admit generalized logical operations was discovered by Lawvere (1969, 1970, 1973). In this section we must

assume a familiarity with category theory, in particular the notion of an adjunction. The reader may consult Maclane (1971) particularly for the less widely known notions of *end* and *coend*. There are axiomatic descriptions of some or all of the formal properties listed here, notably Lawvere's notion of a *hyperdoctrine* (1969, 1970), and Street's notion of a *cosmos* (1974). We shall not attempt to present these; rather, we specify the analogy (first observed by Lawvere) that motivates these general formulations.

The basic form of the analogy is that *types* are interpreted as small categories, *terms* are interpreted as functors between small categories, and *predicates* are interpreted as set-valued functors from small categories. The logical operations which have analogs are characterized by certain *deduction rules*, which, in fact, are seen to be adjunctions. The generalized logical operators for set-valued functors satisfy analogous adjunctions, as follows.

First we consider propositional logic. The truth values for ordinary logic lie in the Boolean algebra 2. Let us write $\rightarrow$ for the order relation; that is, $0 \rightarrow 1$. The truth values of generalized logic are sets.

$$\frac{p \rightarrow q, p \rightarrow r}{p \rightarrow q \wedge r} \qquad \frac{P \rightarrow Q, P \rightarrow R}{P \rightarrow Q \times R}$$

On the left is a familiar deduction rule of propositional logic, and on the right, the adjunction characterizing cartesian product of sets. The counit of the cartesian product adjunction is the transform of the identity function $P \times Q \rightarrow P \times Q$, namely, the pair of projections $P \times Q \rightarrow P$, $P \times Q \rightarrow Q$. These are the analogs of the propositional axioms $p \wedge q \rightarrow p$, $p \wedge q \rightarrow q$. The unit of the cartesian product adjunction is the diagonal function $P \rightarrow P \times P$, analogous to the axiom $p \rightarrow p \wedge p$. Now consider the binary operation $\Rightarrow (p \Rightarrow q = \bar{p} \vee q)$. Its analog is the function set construction $(P, Q \mapsto Q^P)$. The analogous adjunctions are

$$\frac{p \wedge q \rightarrow r}{p \rightarrow (q \Rightarrow r)} \qquad \frac{P \times Q \rightarrow R}{P \rightarrow R^Q}.$$

The units and counits are respectively $p \rightarrow (q \Rightarrow (p \wedge q))$, $P \rightarrow (P \times Q)^Q$; $(q \Rightarrow r) \wedge q \rightarrow r$ (modus ponens), $R^Q \times Q \rightarrow R$ (evaluation). There is also an adjunction characterizing 1 (true) and its analog 1 (singleton set), namely,

$$\frac{\cdot}{p \rightarrow 1} \qquad \frac{\cdot}{P \rightarrow 1},$$

which, for logic, asserts that 1 is maximal, and for sets, asserts that 1 is terminal.

For predicate logic, write $\rightarrow$ for the semantic ordering on predicates; that is, $F \rightarrow G$ means that in all models the interpretation of F is a subset of that of G. Write f, g, etc., for terms. Write Φ, Ψ, etc., for set-valued functors, and ϕ, ψ, etc., for functors between small categories. We have the following analogous adjunctions relating substitution and quantification for logic, and substitution and *Kan extensions* for set-valued functors. Here $\Sigma\phi$ denotes the left Kan extension along ϕ and $\Pi\psi$ denotes the right Kan extension along ψ (see (Maclane, 1971)). Arrows are natural transformations

$$\frac{F(x) \rightarrow G(f(x))}{\exists x(F(x) \wedge [fx = y]) \rightarrow G(y)} \qquad \frac{\Phi(x) \rightarrow \Psi(\phi(x))}{(\Sigma\phi\ \Phi)(y) \rightarrow \Psi(y)}$$

$$\frac{F(g(y)) \rightarrow G(y)}{F(x) \rightarrow \forall y([x = gy] \Rightarrow G(y))} \qquad \frac{\Phi(\psi(y)) \rightarrow \Psi(y)}{\Phi(x) \rightarrow (\Pi\psi\ \Psi)(x)}.$$

If the Kan extensions are written using the appropriate coend and end formulas, we obtain a more striking form of this analogy.

$$\frac{\Phi(x) \rightarrow \Psi(\phi(x))}{\int^x \Phi(x) \times Y[\phi x, y] \rightarrow \Psi(y)} \qquad \text{for} \quad \phi\colon X \rightarrow Y$$

$$\frac{\Phi(\psi(y)) \rightarrow \Psi(y)}{\Phi(x) \rightarrow \int_y (\Psi y)^{X[x,\psi y]}} \qquad \text{for} \quad \psi\colon Y \rightarrow X$$

Finally, for functors $\Phi\colon X^{\mathrm{op}} \times Y \rightarrow \mathrm{Sets}$, $\Psi\colon Y^{\mathrm{op}} \times Z \rightarrow \mathrm{Sets}$, $\Xi\colon X^{\mathrm{op}} \times Z \rightarrow \mathrm{Sets}$, there are analogous adjunctions.

$$\frac{\exists y(F(x, y) \wedge G(y, z)) \rightarrow H(x, z)}{F(x, y) \rightarrow \forall z(G(y, z) \Rightarrow H(x, z))} \qquad \frac{\int^y \Phi(x, y) \times \Psi(y, z) \rightarrow \Xi(x, z)}{\Phi(x, y) \rightarrow \int_z \Xi(x, z)^{\Psi(y,z)}}$$

Indeed, in another context, this amounts to the $\otimes_Y$ — Hom_Z adjunction for bimodules.

The relevance of these observations to interconnection of systems is the following. First, it shows the proper level of generality for the development of Section 5. In fact, by choosing small categories more general than free monoids, one obtains an interconnection theory for other types of system; for example, using Lawvere's algebraic theories, one obtains Thatcher's generalized[2] sequential machines. More importantly, however, it provides a deductive calculus for system homomorphism. A proof from some axioms $F_i \rightarrow G_i$ of some theorem $F \rightarrow G$, using the deductions specified by the above adjunctions, translates as follows. The formulas F_i, G_i, F, G represent interconnections of systems. The proof shows that if one is given certain

homomorphisms $F_i \to G_i$ of systems, there is a canonical induced homomorphism $F \to G$. It is this which makes the theory a *calculus* for system interconnection. A simple use of the last adjunction is given in the next section, and the Kan adjunctions have also been used (Bainbridge, 1972).

Finally, there is an interpretation of the regular expression calculus in this context which provides a special duality principle analogous to the duality theorem of Section 4. For example, if $\Phi: (A + C)^{\mathrm{op}} \times (B + C) \to \mathrm{Sets}$, we may define ${}^{\circ C}\Phi: A^{\mathrm{op}} \times B \to \mathrm{Sets}$ to be $\Phi_C{}^A + \Phi_C{}^A \otimes_C (\Phi_C{}^C)^* \otimes_C \Phi_B{}^C$; where $\Phi_B{}^A$, etc., are the components of Φ; $+$ denotes pointwise disjoint union; $\Gamma \otimes_C \Delta = \int^c \Gamma(\_, c) \times \Delta(c, \_)$; and for $\Psi: C^{\mathrm{op}} \times C \to \mathrm{Sets}$, $\Psi^* = C[\_, \_] + \Psi + \Psi \otimes_C \Psi + \cdots$.

7. Reduction and Minimal Realization for Feedback Systems

The Kan extensions for set-valued functors have been used (Bainbridge, 1972) to describe the minimal realization of behaviors by sequential machines, linear systems, and algebra automata, among others. This can now be generalized to give a minimal realization theory for the behavior of feedback systems. We prove the general result and give its application to sequential machines.

First, for small categories A, B, let us denote a functor of the form $\Phi: A^{\mathrm{op}} \times B \to \mathrm{Sets}$ by $\Phi_B{}^A$. Then, given functors $\Phi_B{}^A$, $\Psi_C{}^B$, $\Xi_C{}^A$, define

$$\Phi_B{}^A \otimes_B \Psi_C{}^B = \int^B \Phi_B{}^A \times \Psi_C{}^B,$$

$$\mathrm{Hom}_C(\Psi_C{}^B, \Xi_C{}^A) = \int_C [\Psi_C{}^B, \Xi_C{}^A],$$

where, for sets P, Q, $[P, Q]$ denotes the function set Q^P. Then $\_ \otimes_B \Psi_C{}^B$ is left adjoint to $\mathrm{Hom}_C(\Psi_C{}^B, \_)$. We can interpret $\Phi_B{}^A$ as $\Phi_{A^{\mathrm{op}}}^{B^{\mathrm{op}}}$, and $\Phi_B{}^A \otimes_B \Psi_C{}^B \simeq \Psi_{B^{\mathrm{op}}}^{C^{\mathrm{op}}} \otimes_{B^{\mathrm{op}}} \Phi_{A^{\mathrm{op}}}^{B^{\mathrm{op}}}$ so $\Phi_B{}^A \otimes_{B\_}$ is left adjoint to $\mathrm{Hom}_{A^{\mathrm{op}}}(\Phi_{A^{\mathrm{op}}}^{B^{\mathrm{op}}}, \_) = \mathrm{Hom}^A(\Phi_B{}^A, \_)$, say.

Reduction Theorem. *Given $\Phi_B{}^A$, $\Psi_C{}^B$, $\Xi_C{}^A$, and a natural transformation $\phi: \Phi_B{}^A \otimes_B \Psi_C{}^B \to \Xi_C{}^A$; there are natural epimorphisms $\alpha: \Phi_B{}^A \to \bar{\Phi}_B{}^A$, $\beta: \Psi_C{}^B \to \bar{\Psi}_C{}^B$ for which there exists a factorization $\phi = (\alpha \otimes_B \beta)\theta$ (where $\theta: \bar{\Phi}_B{}^A \otimes_B \bar{\Psi}_C{}^B \to \Xi$) which is terminal among all factorizations of ϕ through a $\otimes_B$-product of epimorphisms.*

Proof. For simplicity, omit sub- and superscripts, writing Hom. or Hom· as required. Let $\alpha: \Phi \to \bar{\Phi}$ and $\gamma: \bar{\Phi} \to \mathrm{Hom.}(\Psi, \Xi)$ be the image

factorization of the transform $\Phi \to \mathrm{Hom}.(\Psi, \Xi)$ of ϕ. Let $\psi: \bar{\Phi} \otimes \Psi \to \Xi$ be the transform of γ. Performing the dual construction on ψ, we let $\beta: \Psi \to \bar{\Psi}$ and $\delta: \bar{\Psi} \to \mathrm{Hom}\cdot(\bar{\Phi}, \Xi)$ be the image factorization of the transform $\Psi \to \mathrm{Hom}\cdot(\bar{\Phi}, \Xi)$ of ψ, and define $\theta: \bar{\Phi} \otimes \bar{\Psi} \to \Xi$ to be the transform of δ. Then the desired factorization is $\phi = (\alpha \otimes \Psi)(\bar{\Phi} \otimes \beta)\theta = (\alpha \otimes \beta)\theta$.

Now suppose that there is a factorization $\phi = (\xi \otimes \eta)\zeta$, with $\xi: \Phi \to \Gamma$, $\eta: \Psi \to \Delta$ epimorphisms. First, let $\sigma = (\Gamma \otimes \eta)\zeta$, so that $\phi = (\xi \otimes \Psi)\sigma$. Thus, the transform $\Phi \to \mathrm{Hom}.(\Psi, \Xi)$ of ϕ factors through the epimorphism ξ. By the universal property of $\alpha: \Phi \to \bar{\Phi}$, we have $\alpha = \xi\mu$ for some $\mu: \Gamma \to \bar{\Phi}$. The dual construction gives a factorization $\beta = \eta\nu$. Thus $\alpha \otimes \beta = \xi\mu \otimes \eta\nu = (\xi \otimes \eta)(\mu \otimes \nu)$, and since $\xi \otimes \eta$ is an epimorphism, $\mu \otimes \nu$ is uniquely specified. Q.E.D.

MINIMALITY THEOREM. *Given $\phi: \Phi \otimes \Psi \to \Xi$, let $\alpha: \Phi \to \bar{\Phi}$, $\beta: \Psi \to \bar{\Psi}$, $\theta: \bar{\Phi} \otimes \bar{\Psi} \to \Xi$ be as in the reduction theorem. If ϕ factors as $(\xi \otimes \eta)\zeta$ for any maps $\xi: \Phi \to \Gamma$, $\eta: \Psi \to \Delta$, $\zeta: \Gamma \otimes \Delta \to \Xi$, then $\bar{\Phi}$ is a subquotient of Γ, and $\bar{\Psi}$ is a subquotient of Δ.*

Proof. Consider image factorizations $\xi = \bar{\xi}\underline{\xi}$, $\eta = \bar{\eta}\underline{\eta}$. The reduction theorem gives a factorization $\alpha \otimes \beta = (\bar{\xi} \otimes \bar{\eta})(\mu \otimes \nu)$, say, and so we have $\bar{\Phi}$ a subquotient of Γ via $\underline{\xi}$ and μ and $\bar{\Psi}$ a subquotient of Δ via $\underline{\eta}$ and ν. Q.E.D.

As a corollary of the proof of the reduction theorem, we obtain the minimal realization theorem of Bainbridge (1972) for automata and more general systems. As special cases of the reduction theorem we obtain the minimal realization theorem of Bainbridge (1975) for addressed systems, and a minimal realization theorem for feedback systems. For automata, the specializations are the following.

First, for minimal realization of automaton behaviors, let a function $\phi: X^* \to \{0, 1\}$ be given. Let Φ be the left $\varnothing^*$ right X^* action on X^* given by $\epsilon \cdot \xi \cdot \xi' = \xi\xi'$, and let Ψ be the left X^* right $\varnothing^*$ action on X^* given by $\xi' \cdot \xi \cdot \epsilon = \xi'\xi$. Then we may interpret ϕ as a map from $\Phi \otimes_{X^*} \Psi$ to Ξ, the left $\varnothing^*$ right $\varnothing^*$ action on $\{0, 1\}$ given by $\epsilon \cdot p \cdot \epsilon = p$, since $\Phi \otimes_{X^*} \Psi$ is the left $\varnothing^*$ right $\varnothing^*$ action on the set X^*. Thus, we have $\phi: \Phi \otimes_{X^*} \Psi \to \Xi$, and, using half the proof of the reduction theorem, we obtain a factorization $\phi = (\alpha \otimes \Psi)\psi$ for $\psi: \bar{\Phi} \otimes \Psi \to \Xi$. Now $\bar{\Phi}$ is a left $\varnothing^*$ right X^* action on a set $\bar{Q}$, say. Since X^* is free, this action is determined by a function $\delta: \bar{Q} \times X \to \bar{Q}$. The map $\alpha: \Phi \to \bar{\Phi}$ is a homomorphism of left $\varnothing^*$ right X^* actions, and so is determined by its value $\alpha\epsilon \in \bar{Q}$. The left $\varnothing^*$ right $\varnothing^*$ action $\bar{\Phi} \otimes \Psi$ is just the set $\bar{Q}$, so ψ is a function from $\bar{Q}$ to $\{0, 1\}$. Thus, the proof of the reduction theorem gives us an automaton with initial state

$\alpha\epsilon \in \bar{Q}$, transition function $\bar{\delta}: \bar{Q} \times X \to \bar{Q}$ and output function $\psi: \bar{Q} \to \{0, 1\}$. The homomorphism α is the run map, and the factorization shows that the automaton has behavior ϕ. Half the proof of the minimality theorem shows that this is the minimal realization of the behavior ϕ.

Now let us apply the reduction theorem in its entirety to the above ϕ, Φ, Ψ, Ξ. We obtain a factorization $\phi = (\alpha \otimes \beta)\theta$. From $\bar{\Phi}$ we obtain as before a transition function $\bar{\delta}: \bar{Q} \times X \to \bar{Q}$, and from α we obtain an initial state $\alpha\epsilon \in \bar{Q}$. From Ψ we obtain a left action of X^* on a set $\bar{R}$, say, generated by a *cotransition function* $\bar{d}: X \times \bar{R} \to \bar{R}$. From β we obtain an element $\beta\epsilon \in \bar{R}$. Since $\bar{\Phi} \otimes \bar{\Psi}$ is just the left $\varnothing^*$ right $\varnothing^*$ action on $\bar{Q} \times \bar{R}$, θ is a function from $\bar{Q} \times \bar{R}$ to $\{0, 1\}$. However, the function $\hat{\theta}: \bar{Q} \to \{0, 1\}^{\bar{R}}$ induced by θ is a homomorphism of (left $\varnothing^*$) right X^* actions; in other words, $\bar{\delta}$ and $\bar{d}$ satisfy

$$\theta(\bar{\delta}(\bar{q}, x), \bar{r}) = \theta(\bar{q}, \bar{d}(x, \bar{r})) \tag{2}$$

for all $\bar{q} \in \bar{Q}$, $\bar{r} \in \bar{R}$, $x \in X$. Thus, we have an addressed machine (Bainbridge, 1972) with transition function $\bar{\delta}$, initial state $\alpha\epsilon$, cotransition function $\bar{d}$, output coordinate $\beta\epsilon$, and coordinatization θ. The interpretation is this. Each state has, by $\hat{\theta}$, a unique coding as an assignment of binary values to the elements (called *coordinates*) of $\bar{R}$. Equation (2) states that the value stored at coordinate $\bar{r}$ for the next state $\bar{\delta}(\bar{q}, x)$ is the value presently stored at $\bar{d}(x, \bar{r})$. Thus, $\bar{d}$ represents the *information flow* among the coordinates $\bar{R}$. The output of the machine is the function $\bar{Q} \to \{0, 1\}$ obtained by observing the output coordinate $\beta\epsilon$; that is, the output function is $\bar{q} \mapsto \theta(\bar{q}, \beta\epsilon)$. This realization is minimal not only in states $\bar{Q}$, but in coordinates $\bar{R}$.

As a simple but typical feedback system, consider an interconnection of sequential machines having configurations as shown in Fig. 10, where

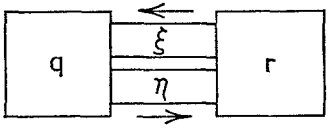

FIG. 10. Configuration of a feedback system.

$\xi \in X^*$, $\eta \in Y^*$, say. A *behavior* function for such a system should assign to each initial configuration (q_0, ξ, η, r_0) a value in some output set, say $\{0, 1\}$; so a behavior function would be of the form $\phi: X^* \times Y^* \to \{0, 1\}$. To see the relation between the machines and such a behavior, suppose the left machine has characteristic biaction Γ, a left $\varnothing^* \times Y^*$ right X^* action

on a set K, and that the right machine has characteristic biaction Δ, a left X^*, right $Y^* \times \varnothing^*$ action on a set L. From Fig. 11, we see that the characteristic

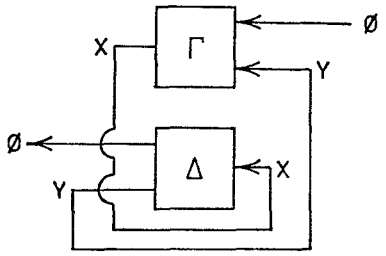

FIG. 11. Reformulation of Fig. 10 interconnection.

biaction of the interconnection is ${}^{\circ Y^{*\mathrm{op}}}({}^{\circ X^*}(\Gamma \times \Delta))$. However, viewing Γ as a left $\varnothing^*$ right $Y^{*\mathrm{op}} \times X^*$ action, and Δ as a left $Y^{*\mathrm{op}} \times X^*$ right $\varnothing^*$ action, one can show from general properties of coends that this is isomorphic to $\Gamma \otimes_{Y^{\mathrm{op}} \times X^*} \Delta$. Specifically, it is the set of equivalence classes of configurations generated by identifying states with their successors; so every configuration may be identified with the cycle of configurations which it ultimately reaches, be that cycle of infinite period, finite period, or period 1 (in effect, a final configuration). In order to assign a behavior to such a system, one must specify initial states k_0, l_0 for the components Γ and Δ, and an output function. By analogy with earlier examples, this should be a map $\zeta: \Gamma \otimes \Delta \to \{0, 1\}$; for example, ζ could have value 1 on equivalence classes corresponding to certain final configurations (cycles of period 1), and 0 otherwise. With these conventions, the machine computes a function $\phi: X^* \times Y^* \to \{0, 1\}$ by first selecting an initial configuration (k_0, ξ, η, l_0), running the machines until ζ can be evaluated (e.g., until the computation halts), then evaluating ζ.

If the systems represented by Γ and Δ are generalized sequential machines (that is, they produce an output string for each input letter, instead of a single output letter) then the tag systems of Post are included as a special case. Since such systems are capable of universal computation, the behaviors can be the characteristic function of an arbitrary recursively enumerable set.

Now, given $\phi: X^* \times Y^* \to \{0, 1\}$, let Φ be the left $\varnothing^*$ right $Y^{*\mathrm{op}} \times X^*$ action on $Y^* \times X^*$ given by $(\eta, \xi) \cdot (\eta', \xi') = (\eta'\eta, \xi\xi')$, and let Ψ be the left $Y^{*\mathrm{op}} \times X^*$ right $\varnothing^*$ action on $Y^* \times X^*$ given by $(\eta', \xi') \cdot (\eta, \xi) = (\eta\eta', \xi'\xi)$. Then $\Phi \otimes \Psi = Y^* \times X^*$. Let $\Xi = \{0, 1\}$. The reduction theorem gives a factorization $\phi: (\alpha \times \beta)\theta$, where $\alpha: \Phi \to \overline{\Phi}$, $\beta: \Psi \to \overline{\Psi}$,

$\theta: \bar{\Phi} \otimes \bar{\Psi} \to \bar{\Xi}$, $\bar{\Phi}$ is a left $\varnothing^*$ right $Y^{*\mathrm{op}} \times X^*$ action on a set $\bar{Q}$, say; and Ψ is a left $Y^{*\mathrm{op}} \times X^*$ action on a set $\bar{R}$, say. Since the actions Φ, Ψ are free, the homomorphisms α, β are determined by their values $\alpha(\epsilon, \epsilon) \in \bar{Q}$, $\beta(\epsilon, \epsilon) \in \bar{R}$. These provide initial states for the Φ and Ψ components of the interconnection. The function θ from $\bar{\Phi} \otimes \bar{\Psi}$ to $\{0, 1\}$ provides the output function. Then $\bar{\Phi}$, $\bar{\Psi}$ with these initial states and output realize the function ϕ in the above sense.

Further investigation is required to determine the precise significance of this minimal realization theorem for closed loop behaviors. The fact that it has the open loop minimal realization theorems as special cases suggests that such investigation is worthwhile.

Before ending this section, it should be noted that the type of feedback interconnection above is not as general as could be considered; for example, a "behavior" could be a homomorphism between systems of the form shown in Fig. 12.

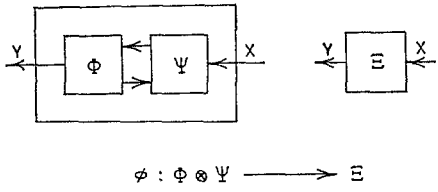

FIG. 12. Most general feedback behavior.

In fact, a *simulation* of Ξ by $\Phi \otimes \Psi$ is a partial map in the topos $\mathrm{Sets}^{Y^{\mathrm{op}} \times X}$ (for Φ, Ψ, Ξ as above), and so corresponds to a map $\phi: \Phi \otimes \Psi \to \tilde{\Xi}$, where $\tilde{\Xi}$ is the classifier of partial maps into Ξ; that is, a behavior to which the reduction theorem applies.

8. LINEAR SYSTEMS

Schützenberger and others have observed that linear systems interconnect according to the calculus of regular expressions. One can find the following manipulations (except the last) in any engineering text on feedback. From Fig. 13, $y = F(x + Gy) = Fx + FGy$; so $y - FGy = (I - FG)y = Fx$; thus, $y = (I - FG)^{-1}Fx$. But $(I - FG)^{-1} = I + FG + (FG)^2 + \cdots = (FG)^*$, so $y = ((FG)^*F)x$.

This section makes two points. First, it explains why, in a paper on a new calculus for feedback interconnection, we do not obtain as a special case the

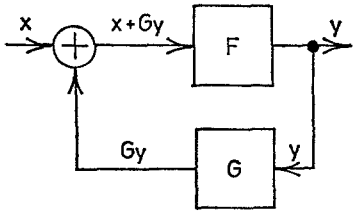

FIG. 13. Linear feedback.

only existing good theory of feedback, namely, that for linear systems. The explanation is that there are two dual notions of feedback, which in the nonlinear case are more appropriately labeled feedback and iteration, but in the linear case may be more easily confused. These two notions admit different calculi, the logical calculus and the regular expression calculus. Feedback in linear systems is described by the regular expression calculus, feedback in *multilinear* systems is described by the generalized logical (i.e., *tensor*) calculus. (For the application of tensor calculus to multilinear systems, see Meseguer and Sols (1975).) The second point is that a unified categorical treatment of both the theory presented in this paper (based on functors Φ: $Y^{op} \times X \to$ Sets) and the linear–multilinear case can presumably be obtained using categories based on closed categories other than Sets.

RECEIVED: January 24, 1975; REVISED: September 5, 1975

REFERENCES

BAINBRIDGE, E. S. (1972), "A Unified Minimal Realization with Duality," Ph. D. dissertation, University of Michigan.

BAINBRIDGE, E. S. (1975), Addressed machines and duality, *in* "Category Theory Applied to Computation and Control" (E. Manes, Ed.), Lecture Notes in Computer Science #25, pp. 93–98, Springer-Verlag, New York.

COHN, P. M. (1965), "Universal Algebra," Harper and Row, New York.

EILENBERG, S. (1971), "Automata, Languages and Machines," Academic Press, New York.

LAWVERE, F. W. (1969), Adjointness in foundations, *Dialectica* **23**, 281–295.

LAWVERE, F. W. (1973), Equality in hyperdoctrines and the comprehension schema as an adjoint functor, *in* "Applications of Categorical Algebra," *Proc. Symp. Pure Math.*, Vol. XVII, pp. 1–14, Amer. Math. Soc., Providence, R. I.

LAWVERE, F. W. (1973), Metric spaces, generalized logic, and closed categories, *Rend. Sem. Mat. Fis. Milano* **43**, 135–166.

MACLANE, S. (1971), "Categories for the Working Mathematician," Springer-Verlag, New York.

MESEGUER, J., AND SOLS, I. (1975), Automata in semimodule categories, *in* "Category Theory Applied to Computation and Control" (E. Manes, Ed.), Lecture Notes in Computer Science # 25, pp. 193–198, Springer-Verlag, New York.

STREET, R. (1974), Elementary cosmoi, *in* "Category Seminar Sydney 1972/73" (G. M. Kelly, Ed.), Lecture Notes in Mathematics # 420, pp. 134–180, Springer-Verlag, New York.

Printed in Belgium