THE UNIVERSITY OF MICHIGAN
COLLEGE OF LITERATURE, SCIENCE, AND THE ARTS
Computer and Communication Sciences Department

Technical Report

HIERARCHICAL DESCRIPTIONS, UNIVERSAL SPACES AND ADAPTIVE SYSTEMS

John H. Holland

ORA Projects 01252 and 08226

supported by:

Department of Health, Education, and Welfare
National Institutes of Health
Grant No. GM-12236-03
Bethesda, Maryland

and

U.S. Army Research Office (Durham)
Grant No. DA-31-124-ARO-D-483
Durham, North Carolina

administered through:

OFFICE OF RESEARCH ADMINISTRATION ANN ARBOR

August 1968

HIERARCHICAL DESCRIPTIONS, UNIVERSAL SPACES AND ADAPTIVE SYSTEMS

John H. Holland

The power of an adaptive system depends critically upon its ability to exploit common factors in successful techniques. If the system has meager means for analyzing elements of its repertory, this ability will be sharply curtailed, no matter how extensive the repertory. Contrariwise, if the system has a great many different ways of describing (or representing) the same device, i.e., if it has a rich variety of ways to decompose elements of its repertory, chances of detecting common factors are greatly enhanced. Each time a device is tried, information accrues about components of each of the potential decompositions. Thus, the richer the variety of decompositions, the higher the effective sampling rate. Of course, to exploit this information about components, the adaptive system must use it to infer the performance of untried devices. And these inferences must, in turn, be used to plan which devices should be generated and tried next. At each stage, the flexibility and success of the process depends upon the flexibility and richness of the system's analysis and synthesis procedures--qualities ultimately depending upon the definitions of structure employed by the system.

In attempting to supply a rich set of representations (with attendant analysis and synthesis procedures), it helps to look at procedures actually exploited by successful adaptive plans. Among the most important are:

(1) Substitution--components common to several highly-rated devices are substituted in other related devices.

(2) Abstraction--by a process of abstraction, highly-rated devices are used to provide schemata (patterns for substitution) for the development of related devices.

(3) Refinement--new "cues" for reacting to the environment are provided by refining the input alphabet or time-scale, adjusting internal processing accordingly.

(4) Modeling--the environment is approximated by some part of the internal structure with the intention of checking predictions of this model against observed outcomes and modifying it accordingly.

(5) Change of Representation--new primitives and operations are introduced so that problems presented by the environment are more easily modeled and related to previous models, outcomes, or stored information.

(6) Metacontrol--rules for employing the preceding techniques are implemented in the device and they in turn are subject to the same techniques; additional levels are added as required.

By searching for structural traits which will give these techniques broad scope, we obtain suggestions for requirements on the structural formalism. These requirements will be briefly described here, and then each in turn will be discussed at length.

1. Hierarchical Description. Substitution and abstraction can be greatly facilitated if the devices used by the adaptive plan have many alternative descriptions in terms of "block diagram" hierarchies. A great variety of schemata can then be formed from a single device by simply deleting the contents of one or more blocks in different ones of its hierarchical descriptions. Any device which satisfies the input-output interface of such an "empty" block becomes a candidate for substitution therein.

Refinement proceeds most easily if alphabets and time-scales of blocks in the hierarchical description can be changed without requiring overall structural reorganization. The minimal constraints possible are those imposed by interfaces with other blocks and by identifications with parts of higher-level blocks.

2. Self-Applicability. The operations for connecting blocks should themselves be defined by exactly the same hierarchical descriptions as the objects to which they are applied (permitting new operations to be introduced as needed). Operations should apply equally directly to blocks at any level in the hierarchical descriptions (permitting any block to be treated as primitive). Taken together these provisions permit the adaptive plan to add new levels of control as required.

3. Incorporation of Models. Primitives and operations should be such that models of the environment can easily be implemented, used, and altered by the overall adaptive plan. There should be a clear means of designating the control exercised by active portions (cf. active subroutines) of the model. There should also be natural structural provisions for making and recording predictions based upon the model. These provisions permit the adaptive plan to check predictions against outcome and make corresponding modifications of responsible parts of the model (a technique used to great advantage by Samuel).

"Some Studies of Machine Learning, Using the Game of Checkers", and "Recent Progress"

Beyond these requirements, there is a further structural consideration which greatly aids study of the interaction of adaptive plan and environment. Because of von Neumann's unfinished work (See Burks' 1967 edited version)

Theory of Self-Reproducing Automata, Edited and completed by A. W. Burks

---

we know that it is possible to represent any mechanistically or computa-

tionally definable adaptive plan/environment combination in a uniform format--a format akin to the "space" of physics with its regular geometry and uniform laws. Such a "space" permits a uniform and rich characterization of both environments and plans. Of greater importance, in this "space" all actions are of the same kind, whether of the plan on itself, or within the environment, or between plan and environment. This and additional advantages will be developed in the discussion of self-applicatility (Section 2).

1. Hierarchical Descriptions

An approach to hierarchical descriptions first requires a notion of the devices or systems to be so described. Discussion here will be limited to systems constructed from finite-state components. That is, admissible components will be those for which relevant behavior can be determined in terms of a finite number of states. The formal counterpart of such a component is the finite automaton defined here by the quintuple

$$a = \langle I, S, O, f, u \rangle$$

where

$$I = \prod_{i=1}^{m} I_i, \text{ where each } I_i \text{ is a finite set}$$

S is a finite set

$$O = \prod_{j=1}^{n} O_j, \text{ where each } O_j \text{ is a finite set}$$

$$f: \quad I \times S \rightarrow S$$

$$u: \quad I \times X \rightarrow O.$$

Under the intended interpretation, I_i is the set of signals possible on the i^{th} input (line) to the component, O_j is the set of signals possible on the j^{th} output, and S is the component's set of internal states. If $I(t) = [I_1(t),\ldots,I_m(t)]$ designates the (ordered) set of signals on the inputs at time t, and S(t) designates the internal state at time t, then the set of output signals at time t, $O(t) = [O_1(t),\ldots,O_n(t)]$, is given by

$$O(t) = u[I(t), S(t)].$$

Similarly, the next state of the automaton is given by

$$S(t+1) = f[I(t), S(t)].$$

A complete input history of the finite automaton, [I(0), I(1),..., I(t),...], is an element of the set I^N of all infinite sequences over I. If we know the initial state S(0) and a complete input history, it follows from the definition of f and u that we can determine the complete sequence of internal states and the complete sequence of output signals, elements of S^N and O^N respectively. That is, f and u can be extended to yield the functions

$$F: \quad I^N \times S \to S^N$$

and

$$U: \quad I^N \times S \to O^N$$

where, formally, $N = \{0,1,2,\ldots\}$ and $I^N = \{\underline{I} : \underline{I}: \quad N \to I\}$, etc.

Our interest will be centered on devices which can be constructed by various means from copies of some intially chosen finite set of automata, $G = \{g_1,\ldots,g_k\}$, which will be called a set of generators (or primitives). Let $\{a_\alpha : \alpha \in A\}$ be a finite or countably infinite set of copies of elements drawn from G and indexed by the ordered set A. The elements of the quintuple corresponding to a_α will be designated I_α, S_α, etc.

We obtain the most easily understood of the systems constructed from the (entire, possibly infinite) collection $\{a_\alpha\}$ if the collection is treated as a single automaton composed of the noninteracting automata a_α. In this automaton the state history or output history of any given component is independent of the histories of all other components. This "free product", or unrestricted composition, A, on $\{a_\alpha\}$ is defined by the quintuple $\langle I_A, S_A, O_A, f_A, u_A \rangle$. Here

$$S_A = \prod_{\alpha \in A} S_\alpha .$$

A second (equivalent) definition of S_A as a collection of functions

$$S_A = \{s: A \to \bigcup_\alpha S_\alpha \text{ for all } \alpha \in A,\ s(\alpha) \in S_\alpha\}$$

provides a convenient notational device; for instance, if s is the state of the automaton $\underline{A}$, then the state of the set of components indexed by $A' \subset A$ is simply the restriction of s to A', $s|A'$. Similarly

$$I_A = \prod_{\alpha \in A} \prod_{i=1}^{m_\alpha} I_{\alpha,i} = \{i: X \to \bigcup_x I_x \text{ } x \in X \text{ and } i(x) \in I_x\}$$

where $X = \{(\alpha,i) \text{ } \alpha \in A \text{ and } 1 \leq i \leq m_\alpha\}$,

and $$O_A = \prod_{\alpha \in A} \prod_{j=1}^{n_\alpha} O_{\alpha,j} = \{o: Y \to \bigcup_y O_y \text{ } y \in Y \text{ and } o(y) \in O_y\}$$

where $$Y = \{(\alpha,j) \text{ } \alpha \in A \text{ and } 1 \leq j \leq n_\alpha\}.$$

$f_A: I_A \times S_A \to S_A$ satisfies the requirement

$$f_A(i,s)\ (\alpha) = s'(\alpha) = f_\alpha[i(\alpha),\ s(\alpha)]$$

where

$$i \in I_A,\ s \text{ and } s' \in S_A, \text{ and } i(\alpha) = [i(\alpha,1),\ldots,i(\alpha,m_\alpha)].$$

That is, f_A when applied to an input state i and an internal state s of $\underline{A}$ yields a new state s'; $i(\alpha)$, $s(\alpha)$, and $s'(\alpha)$ are the values of these states at the component a_α; since the components do not interact we can determine the value of $s'(\alpha)$ from $i(\alpha)$ and $s(\alpha)$ alone, using f_α. Similarly

u_A: $I_A \times S_A \to O_A$ satisfies

$$u_A(i,s)(\alpha) = o(\alpha) = u_\alpha[i(\alpha), s(\alpha)].$$

[FIGURE 1.]

Other systems or compositions over the collection $\{a_\alpha\}$ are obtained by imposing restrictions on the "free product" $\underline{A}$--restrictions resulting from the identification (connection) of selected inputs with selected outputs. Under these restrictions the state or output history of a given component will, in general, depend upon the histories of other components. The identifications determining the restrictions on $\underline{A}$ can be specified by a function γ, the composition function, satisfying the conditions:

(i) γ: $Y' \to X'$ from $Y' \subset Y$, 1-to-1 onto $X' \subset X$;

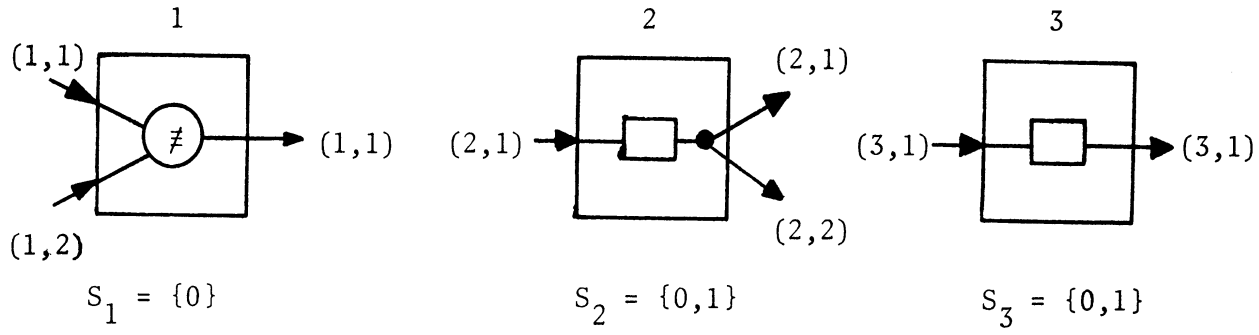
(ii) $O_y \subset I_{\gamma(y)}$ for all $y \in Y'$.

Thus each $y \in Y'$ designates a particular output which is to be connected to a particular input $\gamma(y) \in X'$; (ii) specifies that the output signals be identifiable as signals on the input $\gamma(y)$ -- any encoding or decoding must be carried out by components via their functions f and u. Identification of an input with an output requires the histories of the two to be identical over all time:

$$U_y(\underline{I}_\alpha, s) = \underline{I}_{\gamma(y)},$$

where U_y is the function that gives the output history of $y = (\alpha,j)$ when supplied with the initial state s and the input history $\underline{I}_\alpha$ of the component α to which y belongs. Note that $\underline{I}_\alpha = (\underline{I}_{\alpha,1},\ldots,\underline{I}_{\alpha,m_\alpha})$ and may in turn be constrained by identifications specified by γ. It may

$A = \{1, 2, 3\}$



$S_1 = \{0\}$ $S_2 = \{0,1\}$ $S_3 = \{0,1\}$

$S_A = \{0\} \times \{0,1\} \times \{0,1\}$

$= \{(0,0,0), (0,0,1), (0,1,0), (0,1,1)\}$

$= \{s \to s: \{1,2,3\} \to \{0,1\}\} = \{s_0, s_1, s_2, s_3\}$

where s_2, for example, is given by

α	$s_2(\alpha)$
1	0
2	1
3	0

I_A and O_A are treated similarly, with i_7, for example, given by

(α,j)	$i_7(\alpha,j)$
(1,1)	0
(1,2)	1
(2,1)	1
(3,1)	1

If $S_A(t) = s_2$ and $I_A(t) = i_7$ then

$$S_A(t+1)(1) = f_A(i_7,s_2)(1) = f_1\Big([i_7(1,1),i_7(1,2)],s_2(1)\Big)$$

$$= f_1([0,1],0) = S_1(t+1) = 0$$

and $$S_A(t+1)(2) = f_A(i_7,s_2)(2) = f_2\Big([i_7(2,1)],S_2(2)\Big)$$

$$= f_2([1],1) = S_2(t+1) = 1$$

Figure 1. An Unrestricted Composition.

even be that $\gamma(y) = (\alpha,i)$ in which case the output feeds back directly to one of the inputs of the component, yielding the equation

$$U_{(\alpha,j)}\left((\underline{I}_{\alpha,1},\ldots,\underline{I}_{\alpha,i},\ldots,\underline{I}_{\alpha,M_\alpha}),s\right) = \underline{I}_{\alpha,i} .$$

This amounts to a kind of fixed point requirement on the function U_y, a requirement which may or may not be satisfiable. (See Figure 2.)

[Figure 2.]

In the general case we get a set of simultaneous equations, one for each element of Y', which may or may not be consistent (i.e., satisfiable). Only if the equations are consistent will there exist a quintuple corresponding to the product automaton constrained by γ. The consistency of γ will be assured if it satisfies an appropriate "local effectiveness" condition in addition to conditions (i) and (ii). This condition can be developed along the following lines:

A sequence of output indices

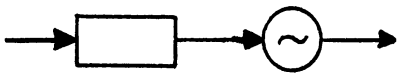
$$y_1 = (\alpha_1,j_1),\ y_2 = (\alpha_2,j_2),\ \ldots,\ y_k = (\alpha_k,j_k)$$

will be called <u>connected</u>, and of length k, if

$$(1 \leq h < k)(\exists b_h)[\gamma(y_h) = (\alpha_{h+1},b_h)].$$

γ will be called consistent with respect to $s \in S_{\underline{A}}$ if, for each $\alpha \in A$, there is an integer $\ell_{s,\alpha}$ such that every connected set of output indices of length $\ell_{s,\alpha}$ which ends at α contains some $y_h = (\delta,j)$ for which $u_{\delta,j}(i,s)$ depends only on unconstrained inputs, i.e.,

The logical net



has the fixed point $\underline{I}$ = 101010... since $U(\underline{I},0)$ = 101010... .

The logical net

has no fixed point since for $\underline{I} = \delta_0\ \delta_1\ \delta_2 \ldots$, $U(\underline{I},0) = \overline{\delta}_0\ \overline{\delta}_1\ \overline{\delta}_2 \ldots$ where $\overline{\delta}_j = 1 - \delta_j$.

Figure 2. Examples of Satisfiable and Unsatisfiable Cases of the Fixed Point Requirement.

$$(i_1, i_2 \in I_A)[i_1 | (X-X') \Rightarrow u_{\delta,j}(i_1(\delta), s(\delta)) =$$

$$u_{\delta,j}(i_2(\delta), s(\delta)].$$

The inputs of the product automaton indexed by the set X - X' are unconstrained by γ and hence may be arbitrarily specified at each time t. For this reason, the set

$$I_{\gamma,A} = \{i | X-X') \ni i \in I_A\}$$

plays the role of the set of input states of the constrained automaton. Similarly the elements of

$$O_{\gamma,A} = \{o | (Y-Y') \ni o \in O_A\}$$

play the role of output states.

Modifying an algorithm of Burks and Wright,

"Theory of Logical Nets"

---

one can prove

Lemma 1.1. If γ is consistent with respect to $s \in S_A$, then there is a unique correspondence $\mu_s : I_{\gamma,A} I_A$ such that, for every i in the range of μ_s and all $y \in Y'$, $u_y(i(\alpha), s) = i(\gamma(y))$. Thus, for any input state of th constrained automaton, when γ is consistent, there exists a unique assignment of states to the constrained inputs $x \in X'$ such that the values $u_{\gamma^{-1}(x)}$, computed for the corresponding constrained outputs $\gamma^{-1}(x)$, are in fact identical as required.

γ will be called a locally effective composition function with respect

to s, abbreviated LECF(s), if (i)γ is consistent with respect to s, and (ii) the consistency of γ with respect to any $s' \in S_A$ implies that γ is consistent with respect to $f_A(\mu_s,(i),s')$ for every $i \in I_{\gamma,A}$.

We obtain as a corollary to Lemma 1.1.

Corollary 1.1. If γ is LECF(s), the correspondence μ_s of Lemma 1.1 can be extended to a unique correspondence μ_s: $I^N_{\gamma,A} \to I^N_A$ such that for every $\underline{I}$ in the range of μ_s and all $y \in Y'$, $U_y(\underline{I}_\alpha, s) = \underline{I}_\gamma(y)$.

For each LECF(s) associated with the product automaton $\underline{A}$, the composition $\underline{A}$ is defined by the quintuple $\langle I_{\gamma,A}, S_{\gamma,A}, O_{\gamma,A}, f_{\gamma,A}, u_{\gamma,A}\rangle$ where $I_{\gamma,A}$ and $O_{\gamma,A}$ are as above

$S_{\gamma,A} = \{s \in S_A \ni \gamma$ is consistent with respect to s$\}$

$f_{\gamma,A}(i,s) = f_A(\mu_s(i),s)$ for $i \in I_{\gamma,A}$ and $s \in S_{\gamma,A}$

$u_{\gamma,A}(i,s) = u_A(\mu_s(i),s)$.

When particular compositions are discussed, the subscript A will be dropped where no confusion can arise. The set of compositions, $\{\underline{A}_\gamma \ni \underline{A}$ is an unrestricted composition and γ is an associated LECF(s) composition function$\}$, will be the set of devices or systems for which we will develop hierarchical descriptions. The set includes representatives of many well-known devices including logical nets, Turing machines with multiple heads and multi-dimensional tapes, and variants of von Neumann's cellular automaton. In the case of devices such as Turing machines and cellular automata, the index set A is countably infinite. When A is infinite the local effectiveness condition assures a critical property--one which permits investigation of finite parts of the overall system with knowledge only of the states of components in a finite "neighborhood" thereof:

Lemma 1.2. Given any $\underline{A}_\gamma$, $\alpha \in A$, and $t \in N$, $S_\gamma(t)|\alpha$ can be calculated from

$S_\gamma(0)|B(\alpha,t)$ and $I_\gamma(0)|B(\alpha,t)$, ..., $I_\gamma(t)|B(\alpha,t)$, where B is a computable function and $B(\alpha,t)$ is a finite subset of A containing α.

We now have a formal definition of the structures which will be candidates for hierarchical description. A given finite composition will have hierarchical descriptions much like a cross-referenced set of increasingly-detailed block diagrams--the kind of diagrams used to describe almost any very complex organization. The highest-level block diagram divides the system into several large, interacting parts. The guiding principle in making this division is a concern that the parts exhibit a "natural" functional coherence in terms of intended use or overall function. It is apparent that different intended uses or different views of the overall function can lead to different highest-level diagrams of the same device. We can thus expect each device to have many distinct hierarchical descriptions. Once the highest-level diagram is set, each of its parts is treated in turn as a system to be further subdivided. The process of subdivision is repeated, generating successive levels of refinement, until parts are reached simple enough not to repay further subdivision. By way of example, a hierarchy of 11 levels in which each block is divided into 10 lower-level blocks would contain 10^{10} lowest-level blocks (about the number of neurons in the human brain). If each of the lowest-level blocks contained a single two-state device, the overall device would have $2^{10^{10}}$ states. Even for a device with as many as 10^{10} components, one need only make a selection at each of 10 levels to uniquely locate any given component. And, assuming a relevant functional division, much will be learned of the effect of that component by observing the use or function of the blocks involved. In contrast, an explicit description of the device in terms of a state transition diagram is not even a possibility; the number of states

involved vastly exceeds the estimated number of atoms in our galaxy. Moreover, presentation in terms of states can be misleading when we study various operations important to adaptation. A state reduction from 1,000,000 to 500,000 states looks impressive, but it may result from the elimination of a single two-state device from a connected set of 20-two-state devices. It may seem impossible to begin with a set of 10^{1000} states and give them any very significant organization and yet, in terms of components, this is achieved almost routinely. Even a small digital computer has a much larger number of states. For devices of this complexity, hierarchical descriptions offer almost the only avenue to detailed understanding.

It will be convenient to define each hierarchical structure formally via a directed tree. Following our previous discussion, each vertex in the tree corresponds to a block in the hierarchical structure. Two vertices, α_1 and α_2, are connected by a directed edge from α_1 to α_2 just in case the block corresponding to α_2 is a part of the block corresponding to α_1. The tree has a root (distinguished vertex) with only outgoing edges, representing the whole device. It also has a set of terminal vertices, representing the blocks at the level of finest detail. Otherwise, vertices have one incoming edge and one or more outgoing edges. The vertices connected to a vertex by its outgoing edges will be called its successor set; each successor set will be ordered. By listing in order the ordinals of vertices in the unique path from the root to any given vertex, we obtain a unique index for each vertex. Thus, the second vertex in the successor set of the root will have the index 1.2, and the first vertex of the successor set of 1.2 will have the index 1.2.1, etc.

We will now examine a simple set of hierarchical descriptions. This

prototype, though simple, adequately illustrates the use of the tree format and will give a concrete background for later discussion. Much more sophisticated descriptions can readily be constructed on the same pattern. Still, the prototype descriptions exhibit the essential property of progressive refinement of the description as levels are added to the hierarchy. That is, as levels are added, the fineness of the time-scale generally increases and constraints are added to the input and output alphabets and to the transition function. From the behavioral viewpoint, this means that an adaptive system employing these descriptions can refine its responses by simply adding levels to the hierarchy describing its current plan. Moreover, in the formation of schemata, the plan can control the range of substitution instances by controlling the level at which blocks are deleted.

Each block in the hierarchical structure will be treated as a composition, although elements of its associated quintuple may be specified only when other parts of the hierarchical structure are specified. The parts of a block may thus be interconnected (by a LECF composition function) leaving only a subset of their inputs (outputs) free. (Recall that the successor set of the vertex associated with a given block designates the parts of that block). The free inputs (outputs) of the parts must be identified, in some order, with the free inputs (outputs) of the block. In effect, subsets of the free inputs (outputs) of the parts will be "cabled" and identified with some free input (output) of the block. Formally, a direct product will be formed of the alphabets of each "cabled" subset (in the order imposed) and this new alphabet will be associated with the designated line of the block. To present this information in precise format labels are attached to each vertex of the directed tree as follows:

Let α be the index of an arbitrary vertex and let $\alpha = \alpha'.k$ so that the vertex is the k^{th} vertex in the successor set of α'. The block associated with α will have m_α inputs and n_α outputs; accordingly the vertex α will be labelled by two vectors,

$$v_\alpha = (v_{\alpha,1},\ldots,v_{\alpha,m_\alpha})$$

$$w_\alpha = (w_{\alpha,1},\ldots,w_{\alpha,n_\alpha})$$

where

$v_{\alpha,j} = (\ell,j_1)$ if $\gamma^{-1}(\alpha,j) = (\alpha'.\ell,\ j_1)$, that is if input j of α is connected to output j_1 of block $\alpha'.\ell$ which is also a part of block α'

$= (j_o)_h$ if $I_{\alpha,j} = \text{proj}_h\ I_{\alpha',j_o}$, that is if input j of block α is component h of a "cable" identified with input j_o of block α'.

$w_{\alpha,j} = (\ell,j_1)$ if $\gamma(\alpha,j) = (\alpha'.\ell,j_1)$

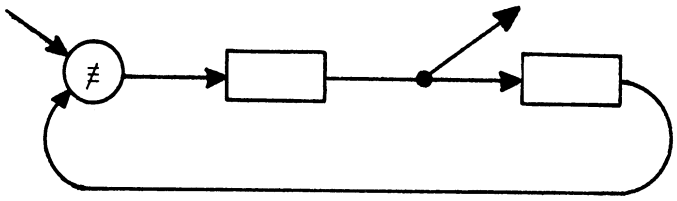
$= (j_o)_h$ if $O_{\alpha,j} = \text{proj}_h\ O_{\alpha',j_o}$.

See Figure 3.

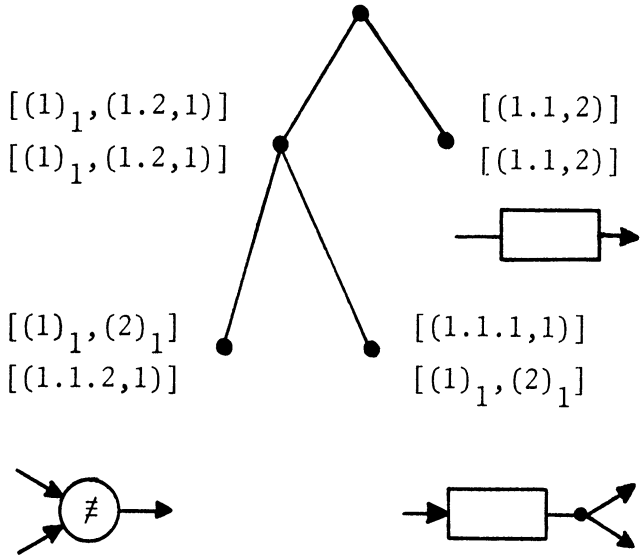
[Figure 3.]

To allow arbitrary relations between the time-scales (the internal clocks) of different blocks--for example, to allow one block to execute several operations on receipt of each signal from another--we must extend this formalism. So that there will be no absolute limit to the refinement of time-scales, it is preferable to specify block time-scales in relation to one another, rather than in relation to some absolute. To accomplish this, let us adopt the convention that the elements of any input alphabet

The logical net



has as one of its hierarchical descriptions the diagram



where inputs and outputs are ordered from top to bottom (in each block) and and vertices are ordered from left to right (at each level).
(Note that cabling and rates are not exemplified).
Deletion of the two lowest level vertices in the hierarchical description yields a schema which can be represented as the incompletely specified logical net.

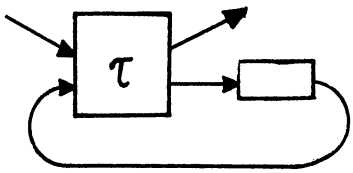


Figure 3. Example of a Hierarchical Description.

$I_{\alpha,j}$ may in fact be a set of strings of some fixed length $k_{\alpha,j}$. Treat output alphabets similarly. Then relax the requirement $O_{\alpha,j} \subseteq I_{\gamma(\alpha,j)}$ to the requirement that, for some r, $[O_{\alpha,j}]^r \subseteq I_{\gamma(\alpha,j)}$ or else $O_{\alpha,j} \subseteq [I_{\gamma(\alpha,j)}]^r$; that is, the elements of $I_{\gamma(\alpha,j)}$ are strings of length $rk_{\alpha,j}$ formed by concatenation of strings in $O_{\alpha,j}$, or vice-versa. With this convention, r time-steps in α will be required to generate a single input symbol for the input line $\gamma(\alpha,j)$ or alternatively, one time-step in α will generate r input symbols for $\gamma(\alpha,j)$. In compositions, relative rate will be indicated by labelling each output (α,j) with the value $\delta(\alpha,j)$, where $\delta(\alpha,j) = r$ when $[O_{\alpha,j}]^r \subseteq I_{\gamma(\alpha,j)}$ and, by convention $\delta(\alpha,j) = \frac{1}{r}$ when $O_{\alpha,j} \subseteq [I_{\gamma(\alpha,j)}]^r$. These values will be indicated in the hierarchical description by augmenting $v_{\alpha,j}$ and $w_{\alpha,j}$; thus

$v_{\alpha,j} = \left((\ell,j_1),\frac{1}{r}\right)$ if $\gamma^{-1}(\alpha,j) = (\alpha'.\ell,j_1)$ and output $\gamma^{-1}(\alpha,j)$ has $\frac{1}{r}$ as its rate indicator, that is $O_{\gamma^{-1}(\alpha,j)} \subseteq [I_{\alpha,j}]^r$

All other cases are treated similarly.

There are rate assignments which make inconsistent requirements on the time-scales of blocks in the hierarchy; however, there exists a convenient necessary and sufficient condition for assuring consistency. It can be developed as follows:

Let $\delta_1, \delta_2, \ldots, \delta_k$ be the rate indicators associated with an arbitrary connected set of output indices $y_1 = (\alpha_1,j_1), \ldots, y_k = (\alpha_k,j_k)$. With each connected sequence we can associate the product $\pi_h \delta_h$. Consistency of time-scales requires:

(i) Any two connected sequences which start at the same block and end at the same block must have the same value for their associated products.

(ii) Any cycle (a connected sequence which begins and ends at the same block) must have the value 1 for its associated product.

These conditions must be satisfied even for connected sequences which are "hidden" (implicit) in "cables".

Consider now a directed tree with labels v_α and w_α, as just defined, assigned to its vertices. The result is a kind of organizational skeleton which can be fleshed out by assigning specific procedures to the terminal vertices. In the present context this is simply accomplished by assigning to a terminal vertex β any composition with the number of inputs and outputs required by v_β and w_β. If one or more terminal vertices in the directed tree are left with no composition assigned, the result will be called a description schema. If all terminal vertices have compositions assigned, then we must determine whether the result in fact describes a composition. To do this we must first determine a composition function from the hierarchical description. This can be done recursively:

Assume that the composition functions $\gamma_{\alpha.1},\ldots,\gamma_{\alpha.b_\alpha}$ have been determined for parts $\alpha.1,\ldots,\alpha.b_\alpha$ of block α. Let (β_1,k_1) be an arbitrary output line identified as component h_1 of line $(\alpha.\ell_1,j_1)$ and let (β_2,k_2) be an arbitrary input line identified as component h_2 of line $(\alpha.\ell_2,j_2)$. Then the composition function for block α is determined by the following condition on pairs of input and output lines. $\gamma_\alpha(\beta_1,k_1) = (\beta_2,k_2)$ if and only if

(i) $\gamma_{\alpha.\ell_1}(\beta_1,k_1) = (\beta_2,k_2)$, i.e., the lines are already connected in part $\alpha.\ell_1$;

(ii) $v_{\alpha.\ell_1} = \left[(\alpha.\ell_2,j_2),r\right]$ and $h_1 = h_2$, i.e., the "cables" containing the lines are connected at this level and the lines occupy corresponding positions in the two cables.

(All lines (β,k) not belonging to the domain or range of γ_α will perforce be identified by some $v_{\alpha.\ell}$ or $w_{\alpha.\ell}$ as components of some line of α). The recursion is started at the terminal vertices by using the composition functions of their assigned compositions. The end-result of the recursion is the composition function γ_1 obtained for the root vertex at the end of the recursion.

Not every composition function derived in this way is LECF. Of course if the description is of an extant finite composition the derived composition function is that of the extant composition. But the description may be obtained by substitution of arbitrary compositions at the terminal vertices of a schema. Then whether or not the derived function is LECF depends upon the particular compositions assigned; if the function is LECF then a composition is indeed described, otherwise not. There are various sufficient conditions for assuring the LECF property. Perhaps the simplest is an addition to requirement (ii) for time-scale consistency: Each cycle must contain an element with delay (an element such that the output function, u, is a function of S only, i.e., u: $S \to O$). Later, when we discuss self-applicability, we will come upon a more important technique for assuring the LECF property. We will see that hierarchical descriptions can be "embedded" in certain compositions, of countably infinite index, generated by a single element--the counterparts of von Neumann's cellular automaton. Moreoever "construction" operations for modifying the descriptions and substituting in schema can be embedded in the same composition. When this is appropriately carried out we will find that no sequence of construction operations can yield a hierarchical description without the LECF property.

The class of hierarchical descriptions just presented, though

simple, has several valuable features. Many of these features follow directly from the definition of the underlying compositions: All "interface" transformations, such as fan-in, fan-out, encoding, decoding, matching of cable components, and change of relative clock rate, are represented explicitly in the descriptions. Thus, fan-out--the sending of the same output signal over several lines--is accomplished by an explicit composition with one input line and several output lines; each output line y has the same associated function, u, which simply places each input signal, i, on the output line, i.e., $u_y(i,s) = u(i,s) = i$, independent of s. Encoding--for example, to transform the output signals of one device to form acceptable by another--is accomplished by a composition which takes as argument the signal to be encoded, o, and yields as value the encoded form, c(o), i.e., u(o,s) = c(o). One device can be required to operate at k times the clock rate of another by using an "interface" composition which has an input alphabet of strings of length k over the output alphabet of the controller and an output alphabet identical to the input alphabet of the controlled device. Similar procedures hold for other interface requirements. The advantage of this explicitness becomes apparent when we wish to compare and manipulate descriptions or the underlying devices. Devices constructed independently can be made compatible, for use in some larger device, by addition of explicit interface devices. More will be said along this line presently.

2. Self-Applicability

In preparation for a discussion of self-applicability of hierarchical descriptions, let us preview the uses of description schemata.

First, note that elimination of any subtree or set of subtrees from a hierarchical description automatically produces a schema; each unassigned vertex so-produced indicates a place for substitution. The substitution instances are of two kinds--those which produce descriptions of compositions, and those which yield new schema. In both cases the substitution instances must satisfy the interface requirements imposed by the remainder of the hierarchical description. In the case of instances of the first kind, these requirements are restrictions on the number, ordering and alphabets of input and output lines, and restrictions on the relative clock rate. The admissible substitution instances of the first kind are the compositions, or henceforth their descriptions, which satisfy the restrictions at the unassigned vertex. Of course, an otherwise unsuitable composition may be modified by interface devices which transform inputs, outputs, and rates, sufficiently to meet the requirements. Substitution instances of the second kind are schemata which meet the interface requirements at the unassigned vertex, insofar as they apply. A substituted schema refines the original schema by the addition of new structural features. The refined schema thus admits only a subset of the substitution instances of the original schema.

The schema's particular significance for adaptive plans emerges when we assign a measure of performance to each device-environment combination. As a result, given any environment from the set of posibilities, each substitution instance of a schema is assigned a numerical value. If now there is a probability distribution over the substitution instances we

can determine an expected value for the (instances of the) schema. The distribution corresponds to a kind of preference ordering on the instances--instances assigned a high probability, by the distribution, being favored. In the context of adaptation, then, this distribution plays a role quite analogous to simplicity orderings in inductive logic. That is, in the usual presentation of inductive logic, there exists an infinite set of incompatible hypotheses agreeing with any finite presentation of evidence. However, the evidence is taken to confirm the hypothesis which is simplest (according to the ordering) and that hypothesis is held (tentatively) until further evidence is gathered. It is as if the hypotheses were "tested" one by one in the order given, each being rejected in turn, until one is encountered which satisifies the evidence. In applied situations, the tentative hypothesis will be a source of predictions (consequences) which will, in turn, influence the actions taken to gather new evidence. In the case of schemata, the distribution over instances (stochastically) sets the order and intensity with which the various schema will be tested. If we think of the schemata as hypotheses about useful organizational principles, then such hypotheses will be tried and (provisionally) "confirmed" or "disconfirmed" in the (stochastic) order imposed by the distribution. That is, as instances of a schema are tested, an estimate can be made of its value; by making the distribution conditional on this estimate, the probability of particular future tests can be altered accordingly. The overall effect is that of "accepting" or "rejecting" the hypotheses corresponding to the schemata, future action being based on the high-valued ("accepted") schemata.

There will be much more to say about the generation of distributions both later in the discussion of self-application and still later in the

discussion of modelling. For now the central point is the possibility of ranking schema according to estimates of their expected performance, onee a distribution is given.

Estimates of the expected performance of given schemata provide an adaptive plan with a natural basis for inferences about untried schemata and devices. These inferences can in turn guide the plan in its selection of devices (and schemata) to be tested next. (If the device selected for test is presented by a hierarchical description, the result of the test provides a sample point for each of the schemata which can be derived from the description). For instance, the plan may take several samples of refined versions of a schema having a high performance estimate relative to other schemata tested. Such refinements are obtained, as indicated earlier, by substituting new schemata at the terminations of the given schema. If the performance measure varies significantly over instances of the given schema, then restriction of samples to an appropriate subset of instances will yield a higher expected performance. That is, there is a reason for testing refinements of the given schema, searching for one which restricts the instances appropriately. The schemata constitute a (highly redundant) covering of the set of devices; the refinements of a given schema constitute a (highly redundant) covering of its instances. In these terms, the object of refinement is to search out elements of the cover progressively converging on devices of high performance. Refinement, however, is only one possible procedure. To give one other example, the plan may "step" from one element of the cover to another by first deleting part of a schema and then substituting a new part. There are many other modification procedures, each appropriate to particular sets of sample outcomes. The procedures to be applied in given situations

must be decided, at a higher level, by the adaptive plan. Recalling our earlier discussion, this amounts to modification of the probability distribution over substitution instances.

How is the adaptive plan to exert the higher-level control implicit in the selection of procedures for selecting (modifying) devices? The introductory remarks suggested a direct route: Make the hierarchical descriptions self-applicable. Then certain of the devices employed by the plan can control the selection of other devices to be tested against the environment. Moreover, additional levels of control can be supplied as needed, in exactly the same way. But how do we enable devices, hierarchically described, to operate on descriptions and schema?

One way to assure self-applicability is to "embed" the adaptive plan, including the descriptions of the devices it employs, in a common "logical space". This possibility is a direct consequence of von Neumann's work on cellular automata. To examine it we need a definition of the "logical spaces" and of the process of "embedding" descriptions in these spaces. The required "logical spaces" can be defined as particular compositions of countably infinite index, generated by a single element. However, in order to distinguish the appropriate compositions, a prior definition of "embedding" is necessary. To specify the manner in which one device (the object) is embedded in (is simulated by) another (the image) is to supply a mapping whereby actions in the image device can be reinterpreted as actions of the object. Since our purpose is to embed detailed descriptions, we wish to preserve in the image not only overall behavior, but also local details of the action. That is, in the image, we wish to find a counterpart of every detail of the object's structure; and we wish to be able to determine from these counterparts everything that could be

determined from the original parts. This can be accomplished for compositions as follows:

Assume composition $\underline{A}_\gamma$ is to be embedded in composition $\underline{D}_\xi$. Then the image of $\underline{A}_\gamma$ under the embedding is to be a subcomposition $\underline{B}_\zeta$ of $\underline{D}_\xi$, where subcomposition is defined by the requirements

(1) $\{b_\beta \ni \beta \in B\} \subset \{d_\delta \ni \delta \in D\}$

(2) $\zeta = \xi|Y_B$, where $Y_B = \{y \in Y' \ni Y'$ is the domain of ξ, $\text{proj}_1 y \in B$, and $\text{proj}_1\, \xi(y) \in B\}$. (When no confusion can arise the subcomposition will simply be denoted $\underline{D}_{\xi|B}$.)

Let A, X, Y and S_γ, I_γ, O_γ be the index and state sets, respectively, of the object composition $\underline{A}_\gamma$ and let B, X_1, Y_1 and S_ζ, I_ζ, O_ζ be the corresponding sets for the image subcomposition $\underline{B}_\zeta$. An <u>embedding</u> will be defined by a mapping ϕ from the sets A, X, Y and S_γ, I_γ, O_γ to subsets of B, X_1, Y_1 and the sets S_ζ, I_ζ, O_ζ, respectively, satisfying:

(1) Distinct {indices, states} map onto distinct {sets of indices, states}:

$$\alpha \neq \alpha_1 \Rightarrow \phi(\alpha) \cap \phi(\alpha_1) = \text{null set, etc.}$$

(2) Each index of a {free, bound} {input, output} of a given element maps onto corresponding indices of the image subset:

$$\text{proj}_1\, \phi(x) \subset \phi(\text{proj}_1 x)$$

$$x \in X\text{-}X' \Rightarrow \phi(x) \subset X_1 - X' \quad \text{and similarly for y}$$

(3) If $x = \gamma(y)$, then the same must hold for all indices in $\phi(x)$:

$$\phi[\gamma(y)] = \zeta[\phi(y)].$$

(4) If an element a_α of $\underline{A}_\gamma$ is assigned the same state by two states of the composition, then the same must hold true in the image:

$$s(\alpha) = s_1(\alpha) \Rightarrow \phi(s)|\phi(\alpha) = \phi(s_1)\ \phi(\alpha)$$

and similarly for $i, i_1 \in I$ and $o, o_1 \in O$.

(5) The transition and the output functions of the image of each element a_α of $\underline{A}_\gamma$ must faithfully represent the transition and output function, respectively, of a_α:

$$f_{\phi(\alpha)}[\phi(i)|\phi(\alpha),\phi(s)|\phi(\alpha)] = \phi f_\alpha[i(\alpha),s(\alpha)],$$

where $f_{\phi(\alpha)}$ is the transition function of the subcomposition indexed by $\phi(\alpha)$, and similarly for $u_{\phi(\alpha)}$.

One final requirement assures that the image subcomposition is immune to disturbances via signals over unassigned inputs in the image. $i \in [I_\zeta|\phi(x) - \phi(I_\gamma)|\phi(x)]$ (an "illegal" signal on an assigned line) is still permitted to cause aberrant behavior:

(6) When the state of $\underline{B}_\zeta$ is the image of a state of $\underline{A}_\gamma$, then f_ζ and u_ζ do not depend upon the states of unassigned inputs:

$$i|\phi(X-X') = i_1|\phi(X-X') \Rightarrow f_\zeta[i,\phi(s)] = f_\zeta[i_1,\phi(s)]$$

where $s \in S_\gamma$ and $i, i_1 \in I_\gamma$, and similarly for u_ζ.

See Figure 4.

[Figure 4.]

It should be noted that, under this definition, several connected elements in the image may be used to represent a single element in the object. We shall have use for a stricter notion: An <u>isomorphic embedding</u> of $\underline{A}_\gamma$ on $\underline{B}_\zeta$ is an embedding such that, for all elements in A, X, Y, S_γ, I_γ, O_γ and B, X_1, Y_1, S_ζ, I_ζ, O_ζ,

$$\alpha \simeq \phi(\alpha),\ x \simeq \phi(x),\ y \simeq \phi(y),\ S_\alpha \simeq S_{\phi(\alpha)},\ I_x \simeq I_{\phi(x)},\ O_y \simeq O_{\phi(y)},$$

where "$\simeq$" indicates set isomorphism.

We can proceed now to define the "logical spaces" suggested by von Neumann's work on self-reproducing automata. Our object is a single composition with two basic properties: (i) "universality", in the sense that any finite

OBJECT $\underline{A}_{\gamma}$

$X' = \{x_{12}, x_{21}, x_{31}\}$ $\qquad$ $Y' = \{y_{11}, y_{22}, y_{31}\}$

y	y_{11}	y_{22}	y_{31}
$\gamma(y)$	x_{21}	x_{31}	x_{12}

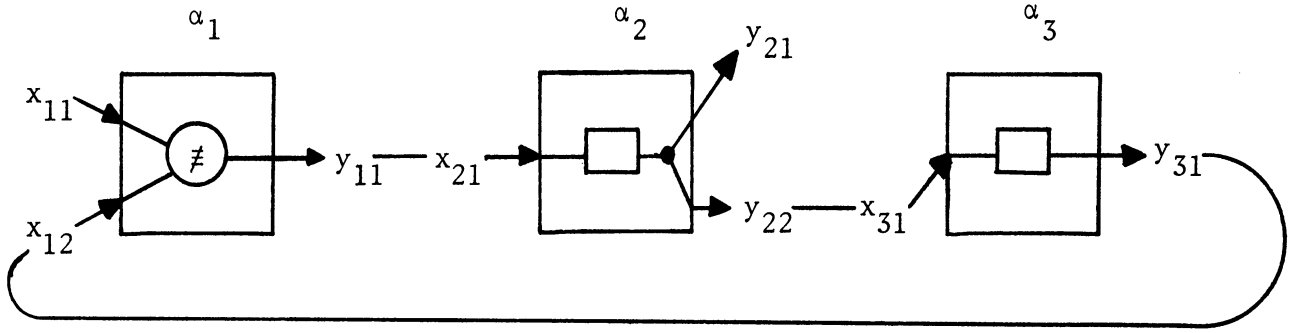
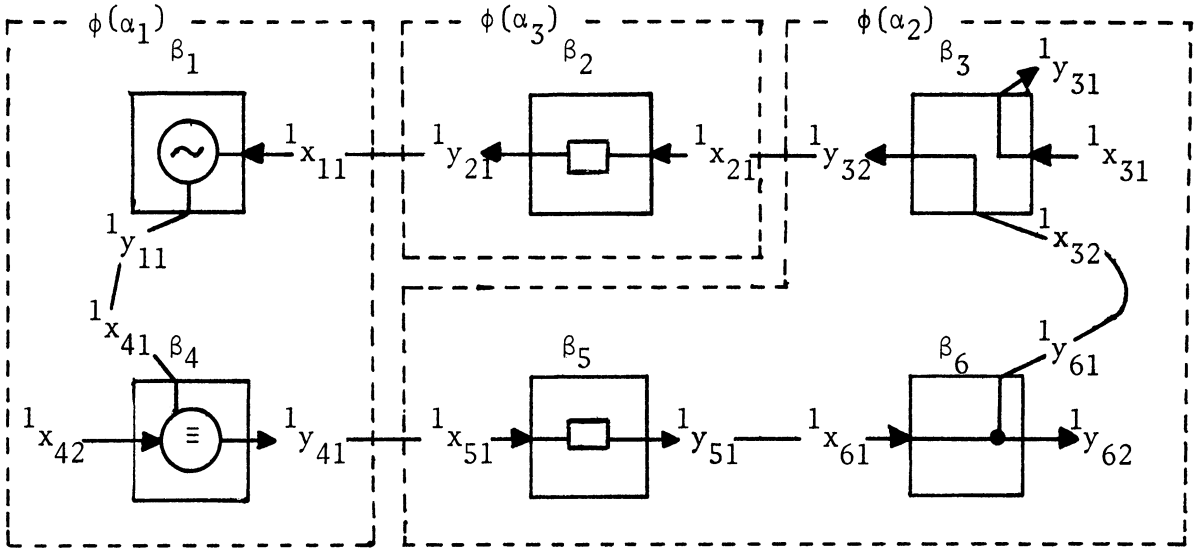


IMAGE $\underline{B}_{\gamma_1}$

$X'_1 = \left\{{}^1x_{11}, {}^1x_{21}, {}^1x_{32}, {}^1x_{41}, {}^1x_{61}\right\}$ $\qquad$ $Y'_1 = \left\{{}^1y_{11}, {}^1y_{21}, {}^1y_{32}, {}^1y_{41}, {}^1y_{51}, {}^1y_{61}\right\}$

${}^1y$	${}^1y_{11}$	${}^1y_{21}$	${}^1y_{32}$	${}^1y_{41}$	${}^1y_{51}$	${}^1y_{61}$
$\gamma_1({}^1y)$	${}^1x_{41}$	${}^1x_{11}$	${}^1x_{21}$	${}^1x_{51}$	${}^1x_{61}$	${}^1x_{32}$



$\phi(\alpha_1) = \{\beta_1, \beta_4\}$ $\qquad$ $\phi(x_{11}) = {}^1x_{42}$ $\qquad$ $\phi(y_{11}) = {}^1y_{41}$ $\qquad$ $\phi(S_{\alpha_1}) \subset S_{\beta_1} \times S_{\beta_4}$

$\phi(\alpha_2) = \{\beta_3, \beta_5, \beta_6\}$ $\qquad$ $\phi(x_{12}) = {}^1x_{11}$ $\qquad$ $\phi(y_{21}) = {}^1y_{62}$ $\qquad$ etc.

$\phi(\alpha_3) = \{\beta_2\}$ $\qquad$ etc. $\qquad$ etc. $\qquad$ etc.

$\text{proj}_1\, \phi(x_{11}) = \text{proj}_1\, {}^1x_{42} = \{\beta_4\} \subset \{\beta_1, \beta_4\} = \phi(\alpha_1) = \phi(\text{proj}_1 x_{11})$, etc.

$\phi[\gamma(y_{11})] = \phi(x_{21}) = {}^1x_{51} = \gamma_1({}^1y_{41}) = \gamma_1[\phi(y_{11})]$, etc.

Figure 4. Example of an Embedding.

composition can be embedded therein, and (ii) "homogeneity", in the sense that, given any two "regions" in the composition, any embedding procedure which works in one region will work in the other. A more precise statement of the latter property can be based upon the notion of one embedding being a "translate" of another:

Given a composition $\underline{D}_\delta$ having a single generator, an embedding ϕ' of $\underline{A}_\gamma$ in $\underline{D}_\delta$ will be called a translate of an embedding θ of $\underline{A}_\gamma$ in $\underline{D}_\delta$ if

(1) There is an isomorphic embedding θ, of the subcomposition indexed by $\phi(A)$ on the subcomposition indexed by $\phi'(A)$, such that $\theta\phi = \phi'$,

(2) for all $\alpha, \alpha' \in A$, any $\xi \in \phi(\alpha)$, $\xi' \in \phi(\alpha')$, and any connected sequence ρ {from ξ to ξ'} {from ξ to $\theta(\xi)$}, there exists a connected sequence ρ' {from $\theta(\xi)$ to $\theta(\xi')$} {from ξ' to $\phi(\xi')$} such that $\text{proj}_2\ \rho = \text{proj}_2\ \rho'$.

A composition $\underline{V}_\nu$ will be called universal and locally homogeneous (for the embedding of finite compositions) or, briefly, universal if:

(1) Given any finite composition $\underline{A}_\gamma$ there is an embedding ϕ of $\underline{A}_\gamma$ into $\underline{V}_\nu$.

(2) If ϕ embeds $\underline{A}_\gamma$ in $\underline{V}_\nu$ then, given arbitrary $\alpha \in A$, $\xi \in \phi(\alpha)$, and $\xi' \in V$, there exists a translate ϕ' embedding $\underline{A}_\gamma$ in $\underline{V}_\nu$ so that $\theta(\xi) = \xi'$; i.e., the same embedding procedure can be used to place the image anywhere within $\underline{V}_\nu$.

A subset of the class of iterative circuit computers

Holland, Iterative Circuit Computers".

---

satisfies the above definition, and hence establishes the existence of

of universal compositions. Certain necessary conditions for a composition to be universal also follow immediately from the definition:

Lemma 2.1. If $\underline{V}_\nu$ is universal then

(1) V must be countably infinite,

(2) $\underline{V}_\nu$ must be generated by a single element g for which the output function u depends properly on both I and S (i.e., the generator is a proper Mealy automaton),

(3) strings over output indices must be "commutative", i.e., if ρ is a connected sequence from α to α', $\sigma = \text{proj}_2\rho$ and σ' is any permutation of σ, then there exists a connected sequence ρ' such that $\text{proj}_2\rho' = \sigma'$.

Part (2) of the lemma follows from the observation that there exist compositions with arbitrarily long connected sequences. The definition of an embedding requires that the image of any such composition contain a connected sequence at least equally long. (Note again that more than behavioral equivalence is required of the image). However, if the output function u_g of the generator g of $\underline{V}_\nu$ depends only upon S, a "delay" is imposed between input and output. As a consequence, in the image, the number of time-steps required to effect the transition function of the connected sequence will depend upon its length, which is unacceptable. Part (3) of the lemma follows from part (2) of the definition of a universal composition. By using a two element image and an appropriate translate of it, one can show that all strings of length two are "commutative"; induction on length then establishes (3).

Corollary 2.1. A universal composition can be "co-ordinatized" so that its elements appear at the intersections of a discrete (integer)

cartesian k-dimensional grid (where $k \leq n_\alpha$, the number of outputs of the generator).

(The proof of the corollary follows easily from part (3) of the lemma.) In terms of this corollary the second requirement in the definition assures that any translation of an image over the grid is also an image of the same object. As a consequence properties of the image, such as its connection scheme, can be made independent of its location in the grid.

While von Neumann's cellular space suggested the class of universal compositions, it is not itself a member of the class. The space is generated by a Moore-type automaton ($u: S \to O$) contradicting condition (2) of the lemma. It might seem that condition (2) could be relaxed enough to admit the von Neumann space if the compositions to be embedded themselves used only Moore-type elements. But this is not so even if a weakened form of embedding, a b-slow embedding, is used. A b-slow embedding lets input signals to the image occur at the reduced rate of once every b time-steps, i.e., at times bt; the states of elements and outputs in the image are then required to occur at the reduced rate of once every b time-steps, after an arbitrary finite initial transient period (which may vary from one composition to another), i.e., at times bt+c.

Theorem. Given any composition $\underline{W}_\omega$ generated by a Moore-type element, any finite set of automata G sufficient to generate all finite automata, and any integer $b \geq 0$, there exist finite compositions $\underline{A}_\gamma$ generated by G which cannot be b-slow embedded in $\underline{W}_\omega$.

Proof outline:

The proof turns on the limited "packing density" of universal spaces, which in von Neumann spaces causes a rapid increase in propagation time.

As a consequence, for object compositions sufficiently large, the transition rate in the image falls behind the input rate.

It is easily established that, if a composition is to be universal for b-slow embeddings, all of the necessary conditions established in Lemma 2.1 apply, with the possible exception of the requirement that the single generator be a Mealy-type automaton. To show that this last requirement also applies, a contradiction will be developed from the assumption that the single generator can be a Moore-type element having a delay between input and output (a 'lag-time') $\tau_g > 0$.

Definition. The separation of two elements in a composition is the length of the shortest connected sequence between them.

Definition. The diameter of a composition is the maximum separation of its elements.

For any Δ there exists an ℓ such that some ('most') compositions of diameter ℓ or greater can only be embedded in a subcomposition of $\underline{W}_\omega$ of diameter at least $\ell + \Delta$.

> Given any set G sufficient to generate all finite automata, there are compositions of diameter ℓ with at least $2^{\ell+1} - 1$ elements. Lemma 2.1 applied to $\underline{W}_\omega$ implies that it can be co-ordinatized by a cartesian grid of some given dimension n. But then less than $(\ell')^n$ distinct elements can belong to any subcomposition of diameter ℓ' in $\underline{W}_\omega$.
>
> Choose ℓ so that $2^{\ell+1} - 1 \geq (\ell+\Delta)^n$.

Among the compositions with the property just described, there exist some (again, 'most') having a cycle which is both functionally dependent upon every composition input and only embeddable in $\underline{W}_\omega$ with diameter greater

than $\ell+\Delta$.

An example is a composition having a cycle of diameter ℓ which includes all the elements in the composition. (See Figure 5.)

[Figure 5.]

The diameter of the cycle's image is $\geq \ell+\Delta$ since there exists a pair of functionally dependent elements in the image of separation $\ell+\Delta$ (because of the diameter of the image subcomposition) and both belong to the cycle.

Because of the functional dependence, the composition can not be embedded as two separate (independent) subcompositions; all elements in the image must be functionally connected.

Let τ_G be the maximum delay ('lag-time') associated with any element of the generating set G and choose $\ell \ni (\ell+\Delta) > \frac{b\ell\tau_G}{\tau_g}$; this condition on ℓ can always be satisfied because ℓ is a logarithmic function of Δ.

Select a composition $\underline{A}_\gamma$ satisfying all the foregoing conditions and in $\underline{A}_\gamma$ select a pair of elements α_1 and α_2 such that the images thereof have a separation at least $\ell+\Delta$. From the set of object sub-cycles containing this pair, select the one which yields the shortest image sub-cycle. α_1 will have (a unique) input x and output y belonging to the connected sequence defining that sub-cycle.

The required functional dependence in $\underline{A}_\gamma$ assures that $I_\gamma(t)|x$ depends in a non-trivial way on $O_\gamma(t')|y$ for some earlier t'. Since the object sub-cycle can be no greater than ℓ in diameter, the total delay from y to x over the connected sequence, $\tau(y,x)$, can be no greater than $\ell\tau_G$. (Briefly, the 'lag-time' between y and x cannot exceed $\ell\tau_G$). Hence $I_\gamma(t)|x$ depends upon $O_\gamma(t')|y$ for some $t' \geq t-\ell\tau_G$

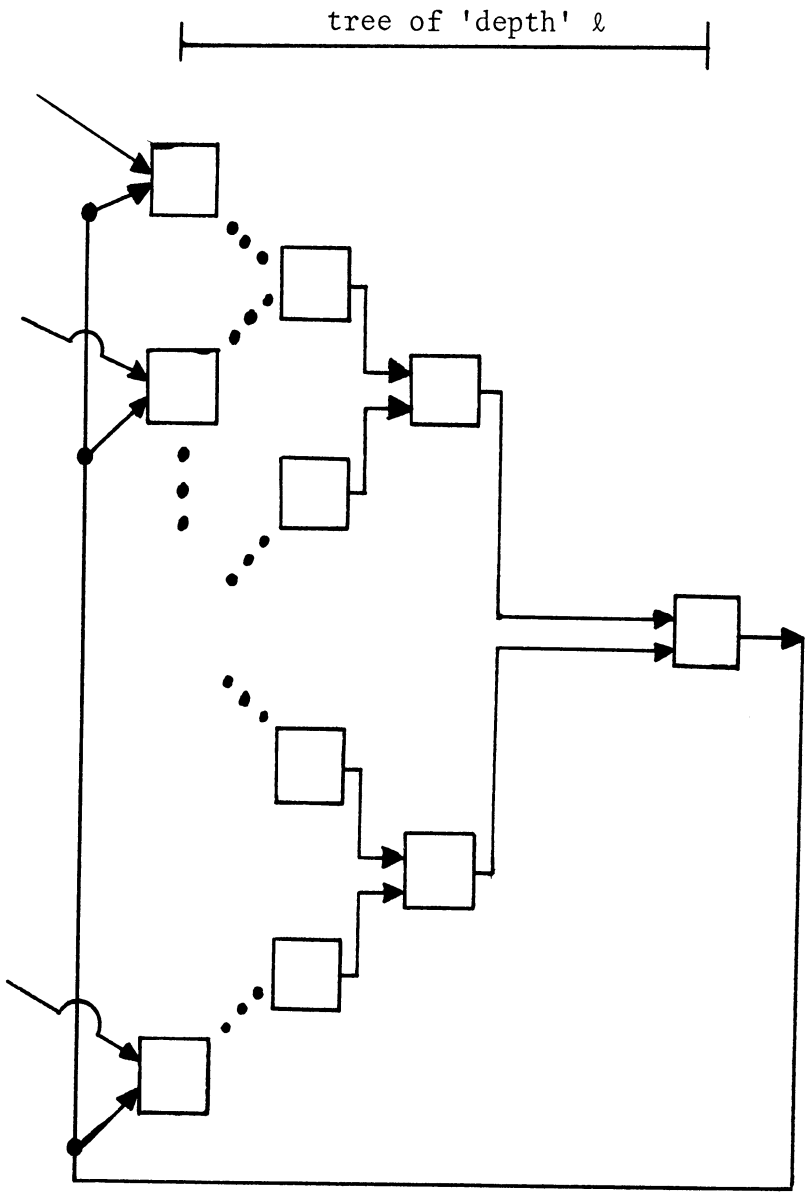


Figure 5. A Composition of $2^{\ell} - 1$ Elements with Cycle of Diameter ℓ.

(i) By the definition of a b-slow embedding, $\phi[I_\gamma](bt+c)\,|\,\phi[x]$ must then depend upon $\phi[O_\gamma](t')\,|\,\phi[y]$ for some $t' = b(t-\ell\tau_G) + c$.

(ii) However, the image subcycle has a diameter $\geq \ell+\Delta$ yielding a total delay $\tau[\phi(y),\phi(x)] \geq (\ell+\Delta)\tau_g$; hence $\phi[I_\gamma](bt+c)\,|\,\phi[x]$ cannot depend upon $\phi[O_\gamma](t')\,|\,\phi[y]$ for any $t' = (bt+c) - (\ell+\Delta)\tau_g > bt+c - b\ell\tau_G = b(t-\ell\tau_G) + c$.

Statements (i) and (ii) contradict each other; hence compositions such as $\underline{A}_\gamma$ cannot be b-slow embedded in $\underline{W}_\omega$. It is not difficult to see that most compositions of sufficiently large diameter satisfy the conditions on $\underline{A}_\gamma$.

———— * ————

This theorem places strong restrictions on the use of von Neumann spaces for studies of self-applicable hierarchical descriptions and higher-level control in adaptive plans. The restrictions are emphasized by the following

Corollary. Given any finite composition $\underline{A}_\gamma$ and any integer $b \geq 0$ such that $\underline{A}_\gamma$ can be b-slow embedded in $\underline{W}_\omega$ as above, 'most' compositions containing $\underline{A}_\gamma$ as a subcomposition cannot be b-slow embedded in $\underline{W}_\omega$.

Thus, given a set of computation procedures with a common "subroutine", it will in general be impossible to embed them in $\underline{W}$ so as to preserve the common subroutine.

The von Neumann space is, however, a member of a broader class, the computation-universal compositions, obtained by weakening condition (1) of the definition of universal compositions:

(1') Given any computable function Γ there is an embedding ϕ into $\underline{V}_\nu$ of a composition $\underline{A}_\gamma$ capable of computing Γ.

See Thatcher, "Notes on Turing Machines and Self-Description and on von Neumann Machines and Self-Reproduction" in Programming Concepts, Automata and Adaptive Systems, for an example of a definition of "computable" tailored to the present context.

In a universal space an embedded device is represented at any moment by a pattern of states assigned to a (usually contiguous) set of elements of $\underline{V}_\nu$. If the object device is a finite automaton, the set of elements will be indexed by a finite subset D of the index set V and will constitute a finite subcomposition $\underline{V}_{\nu|D}$ of $\underline{V}_\nu$. The pattern of states assigned to $\underline{V}_{\nu|D}$ will change to reflect changes of state in the object. But, if no single pattern in the image represents the structure of the object, how are we to modify this structure? The problem is complicated because several quite distinct devices can be embedded in any finite subcomposition of $\underline{V}_\nu$.

Formally, aspects of the structure can be extracted by constructing an appropriate equivalence class over the set of all states $S_{\nu|D}$ of the subcomposition $\underline{V}_{\nu|D}$. Define

$$\sim(s_1,s_2)[f_{\nu|D},I], \text{ for } s_1,s_2 \in S_{\nu|D} \text{ and } I \subseteq I_{\nu|D},$$

if there exist sequences σ_1 and σ_2 over I such that $f_{\nu|D}(\sigma_1,s_1) = f_{\nu|D}(\sigma_2,s_2)$. Then $\equiv(s_1,s_2)[f_{\nu|D},I]$ if and only if $\exists\, s_{j_1}, s_{j_2}, \ldots, s_{j_k} \in S_{\nu|D}$ such that $\sim(s_{j_h}, s_{j_{h+1}})[f_{\nu|D},I]$, for h = 1, 2, ..., k-1, and $s_{j_1} = s_1$, $s_{j_k} = s_2$. If I is the image of the input alphabet of the object device, then a unique element of the equivalence class can be associated with each initialization (initial state assignment) of that device. That is, for any state accessible from the initial state, the associated element of the equivalence class will contain an image state satisfying all conditions for an embedding in $\underline{V}_{\nu|D}$.

Moreover, any initialized device which can be embedded in $\underline{V}_{\nu}|_D$ with I as the image of its input alphabet can similarly be associated with some element of this equivalence class. A change in I will yield a different equivalence relation and a different equivalence class. Given any possible embedding of an initialized device in $\underline{V}_{\nu}|_D$, it can be associated with an element of an appropriately chosen equivalence class. In these terms changes of structure can be associated with transformations between elements of (possibly different) equivalence classes.

How are these transformations to be managed within the space? From the requirements on embedding, we know that changes of state in the image are independent of signals on all input lines to $\underline{V}_{\nu}|_D$ except those lines which are images of the object's lines. Thus, changes of structure must be effected by signals on the image lines, but signals which are not images of object signals. More precisely: Let $\phi(X\text{-}X')$ designate the indices of images of input lines to the object device. For an embedding it is only required that $\phi(I_X) \subseteq I\phi_{(X)}$ for each $x \in X\text{-}X'$. Thus it can easily be arranged that

$$I_c = \prod_{x \in X\text{-}X'} [I_{\phi(x)} - \phi(I_X)]$$

exists and produces desired transformations. Given any embedding in $V_{\nu}|_D$, the set I_c contains the set of all signals capable of modifying the structure of the image. Such a modification may, of course, alter the set of input lines capable of affecting the state transitions in $\underline{V}_{\nu}|_D$--this amounts to a transformation from an element of one equivalence class to an element of another (distinct) equivalence class.

3. Incorporation of Models

From the foregoing it is apparent that the signals from one embedded device may control the construction or modification of another embedded device. It is only necessary that the signals from the controlling device belong to the set I_c of the device to be modified. When this condition is satisfied, the controlling device can control the construction operations in much the same way a finite automaton controls operations on the tape of a Turing machine. Examples of embedded constructing automata can be found in Thatcher's paper referred to on page 36.

See also Codd, Propagation, Computation, and Construction in Two-Dimensional Cellular Spaces.

These devices, because of their essentially serial, step-by-step operations, are (relatively) simple in conception, though not in implementation. Parallel construction procedures, mimicing highly-parallel biological construction such as the chromosome-controlled development of a cell, are also easily conceived and (less easily) implemented.

See Burks, "Computation, Behavior and Structure in Fixed and Growing Automata"; Holland, "Outline for a Logical Theory of Adaptive System,"; and Myhill, "Self-Reproducing Automata" in Programming Concepts, Automata and Adaptive Systems.

There will be more to say about construction procedures shortly, with special reference to hierarchical descriptions, but before that it will pay to take a closer look at the advantages of universal spaces for studies of construction.

When used to study embedded devices, a universal space is an analogue

of the "spaces" used in physics to study its various "mechanics". With a physical space we associate a geometry (e.g. Euclidean geometry) and a set of state-transition laws holding without change at each point in the geometry (e.g. Newton's laws). It is the task of the theoretical physicist to derive the properties of structures embedded in this space (e.g. motions in a central field). From Corollary 2.1 we know that the universal space can be co-ordinatized, thus giving it a (discrete) geometry. By Lemma 2.1 we know that the space must be generated by a single element; thus the transition and output functions of the generator apply at each point, determining behavior accordingly. The result is a kind of discrete physics enabling us to concentrate on the information-processing properties of the embedded devices (in contradistinction to the physicist's primary interest in energy transformations).

One immediate value of this space, mentioned earlier, is an assurance that structures embedded in the space are LECF. Just as the laws of a physical space insure consistency of physical transformations so the transition rules of a universal space insure that any sequence of construction operations executed within the space preserve consistency of the image composition functions. This follows immediately from the requirement that the universal space be a composition--its behavior, and hence that of anything embedded therein, must always be well-defined by the definition of composition. Thus, any sequence of construction operations which can be carried out in the space must transform a device (composition) into a device (composition).

It is a consequence of the geometry of a universal space that only a limited number of modifications and additions can be made per time-step on any given (embedded) device. The limit is set by the number of elements (copies of the space's generator) occupied by the device and potential additions

That is, the amount of construction which can be carried out is dependent upon the size of what is already there--this contrasts sharply with a definition which would permit a construction sequence to be an arbitrary effective (recursive) function over the set of finite automata.

See Wang, "Circuit Synthesis by Solving Sequential Boolean Equations."

Tying the construction process to available resources has important consequences for adaptation. For example, substitution instances for schemata must be drawn, primarily, from a "pool" of already constructed devices or from limited modifications thereof. This requirement amounts to a "simplicity" ordering on the instances tried out by the adaptive plan--an ordering underlying that described earlier. As mentioned there, simplicity orderings play a central role in inductive inference, which is after all the adaptive plan's main task.

The geometry, by assigning each schema to a volume in the (n-dimensional) universal space, has another less desirable consequence. Suitable empty regions must be allocated or prepared to receive the devices which complete the schema. This raises problems of "shape" and "fit" which may be of interest only in quite detailed studies. It is possible to tackle these problems directly; one can set up control devices which, when necessary, change the size of the empty "substitution" regions in the embedded schema, etc.

See Newell, "On Programming a Highly Parallel Machine to be an Intelligent Technician."

However for many purposes, both practical and theoretical, it may be simpler to code the hierarchical descriptions first as a string and then operate thereon. (This is reminiscent of the encoding of three-dimensional protein molecules as one-dimensional strings on the chromosomal helices). "Making room" in a string for the insertion of another string is a much simpler operation than the corresponding operation in two or more dimensions. (The "genetic" operators such as crossover, inversion, etc., provide a ready repertory of string manipulators suited to the present requirements). Any of the standard techniques for embedding a tree structure in a sequential array

---

See McCarthy, "Recursive Functions of Symbolic Expressions and their Computation by Machine, Part I."

---

will serve for reducing the skeleton of the hierarchy to a one-dimensional form. The labels of vertices, other than the terminal ones, are already strings. The terminal instances will either be drawn from the list of primitives or from the pool of descriptions of already constructed devices. Thus if the primitives can be given in string form the whole hierarchical description can be given as a string. This requirement can be met in any of several ways; for example, the primitives can be specified by subroutines for a universal Turing machine or some other general-purpose device.

See Holland, "Outline for a Logical Theory of Adaptive Systems".

(There are of course procedures for making this coding better attuned to genetic operators, but this is a lengthy subject not central to the present

discussion).

If the hierarchical description is encoded in string form, construction becomes a two-step procedure. First the string must be modified to reflect the result of the construction operation. Then the resulting string will have to be read off by an (embedded) "computer" which can either simulate the device described (in the fashion of a universal Turing machine) or else translate the string into an embedded device (in the fashion of a von Neumann constructor). Only in the latter case will there be an immediate test of the modified description to see if it retains the LECF property--if the string can be translated the result is perforce LECF since it lies within the space.

There are two classes of construction operations on hierarchical descriptions: those which rearrange the hierarchical description to obtain a new description of the same device, and those which yield a description of a new device. Among the most important operations of the first type (for the prototype hierarchies given earlier) are those for changing cable configurations (splitting or coalescing cables and reordering wires within cabels) and those for rearranging the hierarchical skeleton (by coalescing levels or distinguishing new levels). Such changes have a strong influence on the schemata likely to be generated and tested by the adaptive plan. Among the most important operations of the second type are those which delete vertices to yield schemata and those which substitute other descriptions at terminal vertices (including devices which act as alphabet or time-scale "translators", modifying interface conditions). The details of these construction operations will of course depend heavily upon the class of hierarchical descriptions and upon the encoding. There are several convenient overall organizations, all more or less routinely implemented. (One of the

most interesting, following a genetic format, would employ hierarchies somewhat different than the earlier prototypes).

Rather than looking at a detailed (and rather routine) example at this point, let us push on to some of the broader questions of implementation and usage.

When discussing substitution in schemata we talked of drawing upon a pool of already constructed devices. Treated properly, this pool can serve as the plan's repository of information about the outcomes and evaluations of previous trials. For instance, the plan can use the pool in such a way that it generates a probability distribution over schemata, a distribution permitting inferences about the performance of schemata (as discussed in Section 2). One way the plan can do this is to treat the descriptions of devices in the pool as a population of individuals undergoing a process of recombination. The plan as applied to the population can then take the following general form:

(1) In preparation for the recombination process, each description is copied a number of times determined by the corresponding device's past performance. That is, each description can be thought of as producing "offspring" in accordance with its performance in the environment confronting the adaptive plan. (As an example, the number of copies may be a random variable having a mean determined by the performance measure).

(2) In the resulting population, descriptions are grouped in sets (singletons, pairs, etc.) appropriate for the application of particular recombination operators. The operators are then applied to these sets to produce the corresponding exchanges and rearrangements of parts.

Of course, this plan could equally well be taken as a general description of the processes of population genetics. Looking to that analogy,

we see that it can easily be arranged that the pool of devices emerging from step (2) has a negligible intersection with the pool of devices presented to step (1). (Barring identical multiple births, no two humans have the same genetic description.) Thus the plan satisfies the desideratum that it be able to proceed where necessary with negligible duplication of trials. At the same time, various schemata will appear multiply, as parts of several descriptions, and will be sampled accordingly. The proportion of a given schema in the overall population will depend upon two factors: the average performance of its instances and its dispersion or generality. (Roughly, the dispersion of a schema increases with the brevity of its hierarchical description. If the string descriptions are formed over a finite alphabet of k letters and, for example, if all combinations of letters are equally likely, then the probability of finding a given schema with m letters in its description will be k^{-m}. While this factor will change as the probability distribution is skewed from uniform, it is clear that the lower the dispersion of a schema the rarer it will be in general.) The net effect of this plan is a time-dependent probability distribution over schemata, conditioned on past performance and generality. Thus, the population summarizes the history of the adaptive plan's confrontation with the environment. In other words, the pool of devices available to the plan at any given time constitutes the current state of its knowledge of the environment, as intended.

If we are to use a universal space to study this plan, the plan must somehow be put into effect within the confines of the space. How is this to be accomplished? It is clear (because a universal space is countably infinite and homogeneous) that any number of devices, acting simultaneously, can be embedded in the space at any time. Thus the pool of devices (and

their descriptions) can be placed in the space.

It is perhaps less clear that arbitrary devices can be embedded so that they can be shifted from point to point without disruption. Yet this seems the most natural way to handle the groupings required by step (2) of the plan (particularly if we want to execute each step of the plan in a parallel rather than a serial fashion). The possibility of a "shift" operation rests on the second requirement in the definition of a universal composition: given two points in the space, and an embedded device arranged around one of the points, there exists a translate of that device similarly arranged around the other point. If the two points are adjacent, then corresponding points in the two images will be directly connected (by outputs of the same index). Now, let the generator of the space be so chosen that the state of the generator can be transmitted over (some) of its outputs (for some $y \in Y'$, $O_y \supseteq S$). If each of the elements constituting the support

---

See Thatcher, "Notes on Turing Machines and Self-Description and on von Neumann Machines and Self-Reproduction" in Programming Concepts, Automata and Adaptive Systems.

---

of the embedded device simultaneously receives a signal causing its state to be transferred over output y, a kind of "shift" operation ensues. The result is a translation of the image within the space. (With some care, it can be arranged that the "shift" signal originates internally when certain conditions are encountered on free inputs of the image subcomposition--the embedded device is then "self-propelled"). Attaining simultaneity of the "shift" signal throughout the image is easily solved in certain iterative

circuit computer spaces

See Holland, "Outline for a Logical Theory of Adaptive Systems".

---

although it leads to a pretty problem in von Neumann's space. (See Moore's discussi

---

See Moore, "The Firing Squad Synchronization Problem."

---

of the "firing squad" problem; the problem arises in any computation universal space which is not universal). Once there is provision for shifting images, it is no great problem to provide the groupings required by step (2) of the proposed adaptive plan.

On this basis we can proceed to implement the remainder of the plan. If the implementation is carried out in an iterative circuit computer, the procedures required can be programmed much as one would write general subroutines for a digital computer. One such version can be set up along the following lines:

(i) The embedded set of descriptions is used to construct the corresponding pool of devices (as outlined on page 41).

(ii) The devices are tested against the environment and the description corresponding to each device is copied a number of times determined by its performance (as outlined on page 43--if performance is determined by payoff and if the number of copies is to be a random variable, then the payoff level is used to set the mean of an associated pseudo-random number generator).

(iii) The descriptions undergo simultaneous random walks within some

region of the space set off by reflecting barriers. (This involves the shift operation discussed on page 45, the direction being controlled by a pseudo-random number generator associated with the description).

(iv) After a period long enough to assure thorough mixing, the descriptions are allowed to pair on contact (but no more) for a period long enough to yield some expected number of pairs. (That is, steps (iii) and (iv) together yield a set of randomly paired descriptions together with a "remainder" set of randomly selected singletons--more could be done, giving the grouping process a much more controlled aspect, but this suffices).

(v) The descriptions, paired and unpaired, undergo recombination; each description has an associated subroutine which (conjointly with its partner, when it is paired) selects (perhaps stochastically) and executes an appropriate recombination operator. (For example, a cross-over-like process could only be executed on pairs--in its simplest form it would amount to aligning the two strings, selecting randomly a length less than the minimal length of the two strings, and exchanging the initial segments of this length. Note that the recombination subroutine could itself be attached as a description to the device description, being used as in step (i) to yield the actual recombination subroutine. The recombination portion of the description could then also undergo recombination, permitting selection of the associated recombination operations on the basis of performance. This procedure could in turn be compounded, automatically, to yield control hierarchies of whatever form proves advantageous vis-a-vis the particular environment confronting the adaptive plan).

(vi) Return to step (i).

There are of course many specific ways of filling in this outline, each replete with an overwhelming amount of detail. In those cases to date in which the details have been filled in, they seem to contribute little to a deeper understanding of the relations between hierarchical structures, cellular spaces, and adaptation. (What understanding is gained is primarily that gained in the working out, rather than in the result; if the working out is a model programmed for simulation, the sumulation itself can be quite suggestive of new paths.)

See Codd, Propagation, Computation, and Construction in Two-Dimensional Cellular Spaces; Rosenberg, Simulation of Genetic Populations with Biochemical Properties; Bagley, The Behavior of Adaptive Systems which Employ Genetic and Correlation Algorithms for relevant results.

I will accordingly once again set aside the burden of detail in order to pursue some more general issues.

Under the foregoing arrangement, information initially supplied to the adaptive plan yields the set of schemata prominent in the initial corpus. These are treated by the plan as primitives out of which to construct better-adapted structures (schemata). Schemata which are successful will persist (occur with substantially higher than average probability) because of a higher than average duplication rate. Thus they have a greater chance of serving as the components of more complex schemata. Generally, the constructed schemata will appear in a significant number of still more complex schemata only if they are more successful than their components. A natural hierarchy emerges. The blocks at the lowest level of the tree are schemata, or instances of schemata, formed by a relatively few operations

on the initial corpus and persistent under successive iterations of the adaptive plan. The higher-level blocks are those, formed in turn by relatively few operations on lower-level blocks, which are also persistent under successive iterations. Without the suggested hierarchical organization, it would take much longer for devices of a similar number of primitives and of comparable success to emerge. For example, assume as on page 44 that some description is at least m letters long on an alphabet of k letters. Then, in the absence of the suggested hierarchical structure, the plan can expect to construct k^m trials before it first encounters the device. Stated in an intuitive but slightly misleading way: There is only time enough for devices with hierarchical organization to emerge under the adaptive plan's guidance. It is useful to compare this hierarchy to the analogous natural hierarchies of stability in open chemical systems, and to the organelle-cell-tissue-organ-organism-species-...hierarchies, paying particular attention to the increase in half-life as one moves up the hierarchy.

Perhaps the most significant feature of this plan is the fact that established schemata, in effect, become new primitives based on the plan's accumulated information about the environment. These new primitives give the plan new ways of representing aspects of the environment. Such changes in representation can be vital in taking advantage of regularities in the environment; indeed such regularities often are revealed only when appropriately represented.

4. Concluding Remarks

The subject of the present discussion is only a preliminary to the investigation of adaptive systems, and it should be recognized as such. Parts of it may have intrinsic formal interest, but in my eyes it will have failed of its objective if it offers no help in resolving questions about adaptation. For such purposes a formal framework and its attendant apparatus are to be tolerated only if they enable answers to be obtained for questions of prior interest--questions originating outside the formalism. In the case of adaptation these questions all center upon the notion of efficiency. At first sight, the concept of efficiency seems far removed from the formal definition of structure, but some of the connections are indicated in the last few paragraphs above. ("Persistence" and "enough time ... to emerge" are concomitants of efficiency.) Answers to similar questions were also von Neumann's ultimate objective in his unfinished work: "...can the construction of automata by automata progress from simpler types to increasingly complicated types? Also, assuming some suitable definition of 'efficiency', can this evolution go from less efficient to more efficient automata?"

---

von Neumann, Theory of Self-Reproducing Automata.

Here, as elsewhere, the overall objective is a formalism sufficiently oriented to reality to permit reliable inference, at least qualitatively, about what would happen in the more complex real situations. (The situation is quite like that of the use of "free fall", in Newtonian physics, as a guide to more complex cases involving, say, atmospheric friction). When so conceived, a formalism has striking advantages as a tool augmenting

empirical investigation. Unlike the real situation, actions within the formal system are completely defined and available for deductive analysis or simulation. The universal spaces are examples par excellence, yielding completely defined universes within which one can embed models of adaptive processes. This largely eliminates the great pitfall of informal theories: placing too great a burden on some vague or ill-defined mechanism which cannot bear the brunt (e.g. a mechanism which cannot possibly act consistently as required). Once an adaptive plan is presented within the formalism its consistency is assured, along with its formal existence, and one can test inferences about it under conditions admitting of no hidden confounding factors.

REFERENCES

1. Burks, A. W., "Computation, Behavior and Structure in Fixed and Growing Automata" in Self-Organizing Systems, Pergamon Press, 1960, p. 282-311.

2. Burks, A. W. and Wright, J. B., "Theory of Logical Nets", Proc. IRE 41, 1953, 1357-1365.

3. Bagley, J. E., The Behavior of Adaptive Systems which Employ Genetic and Correlation Algorithms, The University of Michigan Ph.D. Dissertation, 1967.

4. Codd, E. F., Propagation, Computation, and Construction in Two-Dimensional Cellular Spaces, The University of Michigan Ph.D. Dissertation, 1967.

5. Holland, J. H., "Iterative Circuit Computers", Proc. Western Joint Computer Conference 1960, 259-265.

6. Holland, J. H., "Outline for a Logical Theory of Adaptive Systems", J. Assoc. Computing Machinery 9, 1962, 297-314.

7. Holland, J. H., "Universal Spaces: A Basis for Studies of Adaptation" in Automata Theory, Academic Press, 1966, p. 218-230.

8. McCarthy, J., "Recursive Functions of Symbolic Expressions and their Computation by Machine, Part I", Comm. Assoc. Computing Machinery 3, 1960, 184-195.

9. Moore, E. F., "The Firing Squad Synchronization Problem" in Sequential Machines: Selected Papers, Addison-Wesley, 1964, p. 236-237.

10. Myhill, J., "Self-Reproducing Automata" in Programming Concepts, Automata, and Adaptive Systems, The University of Michigan Summer Conferences, 1966.

11. Newell, A., "On Programming a Highly Parallel Machine to be an Intelligent Technician", Proc. Western Joint Computer Conference, 1960, 267-282.

12. Putnam, H., Probability and Confirmation, Forum Lectures, Voice of America, U.S.I.A.

13. Rosenberg, R. S., Simulation of Genetic Populations with Biochemical Properties, The University of Michigan Ph.D. Dissertation, 1967.

14. Samuel, A. L., "Some Studies of Machine Learning, Using the Game of Checkers I and II-Recent progress", IBM J. Res. and Dev. 3, 211-229 and 11, 601-617, 1959 and 1967.

15. Thatcher, J. W., "Notes on Turing Machines and Self-Description and on von Neumann Machines and Self-Reproduction" in Programming Concepts, Automata and Adaptive Systems, The University of Michigan Summer Conferences 1966.

16. Von Neumann, J., Theory of Self-Reproducing Automata (edited and completed by A. W. Burks), University of Illinois Press, 1966.

17. Wang, H., "Circuit Synthesis by Solving Sequential Boolean Equations", Zeitschift f. Math. Logic u. Grundlagen d. Math. 5, 1959, 391-322.

UNCLASSIFIED
Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Logic of Computers Group The University of Michigan Ann Arbor, Michigan 48104	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

HIERARCHICAL DESCRIPTIONS, UNIVERSAL SPACES AND ADAPTIVE SYSTEMS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Technical Report

5. AUTHOR(S) (Last name, first name, initial)

Holland, John H.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
August 1968	54	17

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DA-31-124-ARO-D-483 b. PROJECT NO.	08226-4-T
c. d.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. AVAILABILITY/LIMITATION NOTICES

Distribution of This Document is Unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	U,S. Army Research Office (Durham) Durham, North Carolina

13. ABSTRACT The power of an adaptive system depends critically upon its ability to exploit common factors in successful techniques. If the system has meager means for analyzing elements of its repertory, this ability will be sharply curtailed, no matter how extensive the repertory. Contrariwise, if the system has a great many different ways of describing (or representing) the same device, i.e., if it has a rich variety of ways to decompose elements of its repertory, chances of detecting common factors are greatly enhanced. Each time a device is tried, information accrues about components of each of the potential decompositions. Thus, the richer the variety of decompositions, the higher the effective sampling rate. Of course, to exploit this information about components, the adaptive system must use it to infer the performance of untried devices. And these inferences must, in turn, be used to plan which devices should be generated and tried next. At each stage, the flexibility and success of the process depends upon the flexibility and richness of the system's analysis and synthesis procedures--qualities ultimately depending upon the definitions of structure employed by the system. Among the many important procedures are those of: Substitution, Abstraction, Refinement, Modeling, Change of Representation, and Metacontrol. A structural formalism well-attuned to such procedures will exhibit three critical characteristics: Hierarchical Description, Self-Applicability, and Incorporation of Models. Formalisms with these characteristics are studied usine a broad class of automaton representations, the class of compositions, which includes countably infinite devices such as von Neumann's cellular space and iterative circuit computers.

DD FORM 1 JAN 64 1473

UNCLASSIFIED
Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Adaptive Systems AutomatatonTheory Cellular Machines Artificial Intelligence						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2*a*. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2*b*. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7*a*. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7*b*. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8*a*. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8*b*, 8*c*, & 8*d*. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9*a*. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9*b*. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through

_____."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through

_____."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through

_____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS), (S), (C), or (U)*

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.

GPO 886-551