W. E. Donath

Equivalence of Memory to "Random Logic"

Abstract: A model of the design process for computer logic is used to estimate the number of bits of memory required to replace a so-called "random logic" circuit. The model can also be used to compare the respective time delays of array logic and random logic.

Introduction

Normally, most of the logic in computers is in the form of so-called "random logic," which consists of electronic circuits performing NOR, NAND, OR, AND or other functions interconnected in a pattern that does not follow some prior scheme. It has been customary for some time, however, to implement the major portion of the control logic in read-only storage, which permits far more flexible machine design. Recently, work has progressed on so-called "array logic," which is essentially a type of look-up logic and, unlike memory, does not require that all possible input combinations be decoded. Either read-only, read-mostly, or read and write storage can be used for array logic.

The purpose of this paper is to point out some combinational results relating random logic to memory. In particular, we point out that a model of a design process developed earlier [1, 2] shows that the number of distinct computers that can be built with C circuit-type elements is of the order of $2^{8.5\,C}$, which we interpret to mean that a circuit is the equivalent of 8.5 cells of memory.

A basic question concerns the number of bits of memory required to replace one circuit in random logic; empirical studies (e.g., [3]) found this "equivalence ratio" to vary between 5:1 and 30:1. It was also observed [3] that the use of larger memory arrays has led to more inefficient use of the memory array. From the point of view of the present study, it is important that we now establish that there exists such a ratio, and that it is basically around 8.5. It is, of course, likely that selected functions may achieve far better ratios of utilization; on the other hand, we do not study here the dependence on array size, which is an open problem.

Another implication of the above result is that the provision of several types of circuits, e.g., if AND and NOR are provided simultaneously, will allow only a modest improvement in circuit count. The equivalence ratio, instead of being 8.5, would then be at most 9.5, which would yield a 12 percent improvement in circuit count.

In the next section we discuss a model of the design process, from which we derive an equation showing that the information content of a logic design is asymptotically proportional to the number of circuits in the design. In the third section we derive actual values from this model.

In the last section we give some consideration to time delay. In essence, we derive a crude and simple rule for comparing the time delay of a memory array with that of a circuit. Here we use previous results [1, 2] on path lengths in computer logic.

Model of the design process

The essential idea in the model is that we use an expansion process that preserves "self-similarity" [4]: Given a graph representation of the computer at a fairly high level of design, we replace each node of the graph by a small graph, and each edge by a set of edges between the nodes of the two graphs replacing the original nodes, as indicated in Figs. 1 and 2. Note that the mathematical basis for many derivations given here comes from the theory of Markov processes [5].

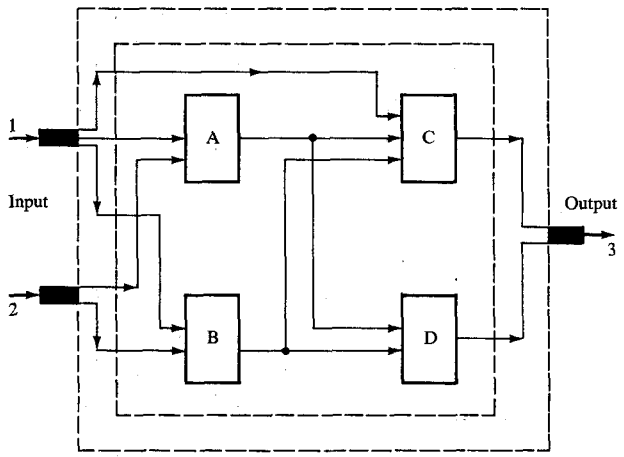

Figure 1 The four blocks, or nodes, represent a pattern to be used in the hierarchical design process for random logic. The outermost outline represents the block in the higher level of the design process. In between the two dotted lines, the input and output edges are replaced by sets of "descendant" edges (see text), where output edge 3 yields two descendant edges and input edges 1 and 2 give rise to three and two descendant edges, respectively. Inside the inner dotted line are four nodes connected to each other with four edges as well as to the expanded external edges.

The following parameters are of interest:

- q the average number of nodes in a replacement graph
- $\bar{s}$ the average number of edges in a replacement edge set
- A the average number of connections to a node at some level of design
- l the number of design levels
- T the average number of connections to a set of nodes, which are the descendants of a single node after l design levels
- C the total expected number of descendant nodes of this single original node.

From these parameters we now define

$$T = A\bar{s}^{l} \tag{1}$$

and

$$C = q^{l}. \tag{2}$$

We relate T to C as

$$l = \log C / \log q$$

$$T = A\bar{s}^{\log C/\log q}$$

$$= AC^{\log \bar{s}/\log q}$$

$$= AC^{p}, \tag{3}$$

where we define

$$p = \log \bar{s}/\log q. \tag{4}$$

The "Rent exponent" of Eq. (3) was originally observed in computer operations by E. Rent [6]; a thorough study of this relationship is that of Landman and Russo [7], where the range of p was found to vary from 0.47 to 0.75. A theoretical basis for this relationship is given in more detail in earlier work [1]; we give this derivation in Appendix A. The starting point of this model was, of course, the need for an understanding of the Rent rule.

We are now concerned with the number of computers that can be designed with such a process. Suppose that w different graphs can be substituted for a node at each stage of substitution, and that there exist no two equivalent nodes at any stage in the process. (The general mathematical problem becomes far more difficult when this is the case, and we expect that the approximation made here is not seriously in error.) Finally, let us denote by W_l the number of distinct graphs generated by the design process after l levels of design. It is obvious that

$$W_l = w^{q^{l-1}+q^{l-2}+\cdots+1}$$

$$= w^{(q^l-1)/(q-1)} \tag{5}$$

Let us assume that

$$C = q^{l} \gg 1,$$

$$W_l = w^{C/(q-1)}$$

and we can then denote

$$\log_2 W_l = C(\log_2 w)/(q-1). \tag{6}$$

We may consider the quantity $\log_2 w/(q-1)$ to represent the information content of a design per elemental node. This simple result shows that the information content of a logic design is proportional to the number of nodes, and we therefore can speak of an "information content" per node. In the next section a value is derived for this quantity, which we find to be about 8.5 bits/node.

We find here for this design process a result that states that the information content of a graph increases linearly with the number of nodes in the graph. In contrast, we may note that the number of directed graphs with C nodes and hC edges (h finite, but $h > 1$) is far larger than that given by Eq. (5) and has an asymptotic behavior quite different from that given by Eq. (6). In the latter case

$$W = \binom{C(C-1)}{Ch}/C!,$$

which becomes, after substituting Stirling's approximation for the large factorials involved, and after some further manipulation,

$$W = C^{C(h-1)}\,(e/h^{h})^{C},$$

or

$$\log_2 W = C[(h-1)\log_2 C + \log_2 e - h\log_2 h].$$

Obviously, the asymptotic behavior is $C \log C$. Note that the design process modeled here generates only a small fraction of all the graphs possible. One might conjecture that most such graphs are too complex to be worked with by human beings.

Evaluation of information content

To perform the exact computation, we need to model the process in somewhat more detail. Each step of the expansion process consists of

1. Replacing edges by sets of edges.
2. Replacing nodes by small graphs.
3. Connecting the edges generated in Step 1 to the nodes of the graphs generated in Step 2.

Later we amplify the design process; first, however, we require some results very similar to Shannon's in information theory. Let us define a set $\{b\}$ of B boxes, where each box b holds N_b elements and each element in $\{b\}$ has weight m_b. We make M choices with replacement, where each of the M choices is distinct from the others (i.e., choosing element x at choice i is not the same event as choosing element x at choice j). Let us fix M_b as the number of times we choose from box b, giving us a total of

$$N(M_1\, M_2 \cdots M_B) = M! \prod_b N_b^{M_b}/M_b! \tag{7}$$

distinct choices (i.e., we do not say a priori if choice i is to be from some prespecified box b). We define the frequency c_b of making choices from box b as

$$c_b = M_b/M. \tag{8}$$

Theorem: For M much larger than B, the number N of distinct choices becomes

$$N = 2^{MH+O(B \log M)}, \tag{9}$$

where the information function H is defined usually as

$$H = \sum_b c_b \log_2 (N_b/c_b). \tag{10}$$

Proof of Eq. (9): M and M_b are large numbers, and we can substitute Stirling's approximation $K! = \sqrt{2\pi K}\,(K/e)^n$ for the factorials in Eq. (7) to give

$$N \approx \sqrt{2\pi M}\, M^M \prod_b \left(\frac{N_b}{M_b}\right)^{M_b} \Big/ \sqrt{2\pi M_b}.$$

Using $\Sigma M_b = M$, we have

$$N \approx \sqrt{2\pi M} \prod_b \left(\frac{N_b M}{M_b}\right)^{M_b} \Big/ \sqrt{2\pi M_b}.$$

Taking the logarithm to the base 2, we find

$$\log_2 N = \sum_b M_b \log_2 (N_b M/M_b) + \log_2 \sqrt{2\pi M} - \sum_b \log_2 \sqrt{2\pi M_b}.$$

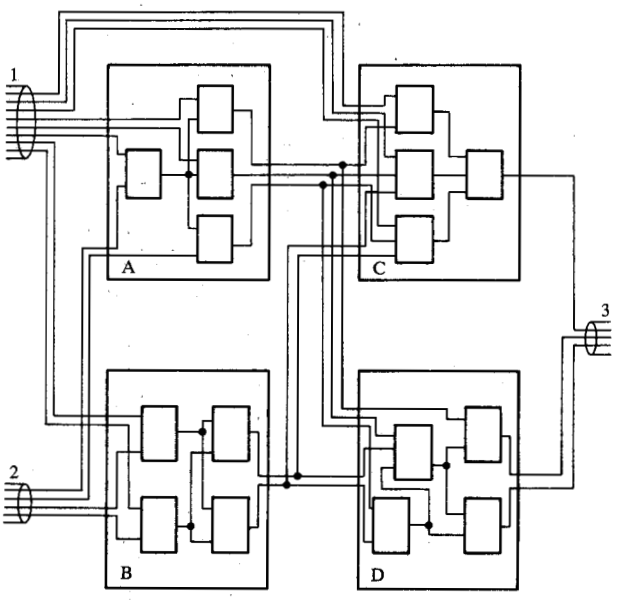

Figure 2 The nodes A, B, C, and D, of Figure 1, expanded one step further in the design process.

Substituting Eq. (8) yields

$$\log_2 N = M \Sigma\, c_b \log_2 (N_b/c_b) + \log_2 \sqrt{2\pi M} - \sum_b \log_2 \sqrt{2\pi M_b},$$

where the first sum is MH. Because $M_b \leq M$ and because there are at most B terms, this simplifies to

$$\log_2 N = MH + O(B \log M). \qquad \text{Q.E.D.}$$

Next we fix the values of M_b so that the average weight $\overline{m}$ of our choices $\overline{m} = \Sigma_b\, m_b M_b/M$ has some predetermined value, while the number of choices N is maximized. We find then that

$$c_b = N_b\, x y^{m_b}, \tag{11}$$

where x and y are parameters fixed so that

$$\Sigma\, c_b = 1$$

$$\Sigma\, m_b\, c_b = \overline{m}. \tag{12}$$

Proof: It is essentially necessary to maximize H. We do this using the method of Lagrangian multipliers, i.e., we maximize with respect to c_b

$$(\ln 2) \times \Sigma\, c_b \log_2 (N_b/c_b) + \lambda(\Sigma\, c_b - 1) + \mu(\Sigma\, m_b\, c_b - \overline{m}),$$

where we introduce the factor ln 2 only for convenience. Differentiating with respect to c_b, we find

$$\log_2 N_b/c_b - 1 + \lambda + \mu m_b = 0,$$

Table 1 Comparison of exact and approximate formulas for number of graphs

q	n	*Exact value*[a]	*Approximate formula* $\binom{q(q-1)/2}{n}/q!$	*Ratio*
6	7	24	9	2.7
7	9	131	38	2.3
8	12	1312	754	1.74
9	14	15615	6600	2.6

[a] J. Riordan, "*An Introduction to Combinatorial Analysis*," J. Wiley & Sons, Inc., New York, 1958, p 196.

which gives us

$$c_b = N_b\, e^{\lambda-1+\mu m_b}.$$

If we set $x = e^{\lambda-1}$ and $y = e^{\mu}$, we obtain Eq. (11). Q.E.D.

Furthermore, we can see that

$$\begin{aligned} H &= \Sigma\, c_b \log_2 N_b/c_b \\ &= -\Sigma\, c_b(\log_2 x + m \log_2 y) \\ &= -\log_2 x - \overline{m} \log_2 y. \end{aligned} \tag{13}$$

For the special case that b assumes the values 0 to B with

$$N_b = \binom{B}{b},$$

$$m_b = b, \tag{14}$$

we find

$$x = [(B - \overline{m})/B]^B,$$

$$y = \overline{m}/(B - \overline{m}),$$

$$H = \overline{m} \log_2 B/\overline{m} + (B - \overline{m}) \log_2 B/(B - \overline{m}). \tag{15}$$

Proof

$$c_b = \binom{B}{b} xy^b$$

$$1 = \Sigma\, c_b = \sum_b \binom{B}{b} xy^b = x(1 + y)^B,$$

$$\begin{aligned} \overline{m} = \Sigma\, bc_b &= \sum_b \binom{B}{b} b\, xy^b = xy\, B \sum_{b>0} \binom{B-1}{b-1} xy^{b-1} \\ &= xyB(1 + y)^{B-1}. \end{aligned}$$

We then find, by dividing the second of the above two equations by the first, that $\overline{m} = yB/(1 + y)$, or $y = \overline{m}/(B - \overline{m})$.

Substituting y into either of the above two equations yields x. Substituting x and y into Eq. (13) yields us the expression for H in Eq. (15).

The question of labeling of the nodes is of some importance; in the derivation of the information content per node [Eq. (6)] we assumed after each step of expansion that the nodes of the parent graph are nonisomorphic to one another. Of course, this is an approximation which would give us too large a count for the number of graphs generated. In this section we assume that the small descendant subgraphs have no isomorphic sets of nodes, which would be an approximation leading to lower counts.

To estimate roughly the amount of error introduced by this approximation, we compare for simple line graphs (i.e., q unlabeled nodes, n undirected edges) the exact count and the approximate formula in Table 1.

It can be noted that approximate values are off by a factor of about two. If w is in error by a factor of, let us say, eight, for $q = 6$, this would introduce an error in the information content per node of, using Eq. (6),

$$\log_2 8/5 = 0.6 \tag{16}$$

as compared to a computed value of 8.5.

We may look at the expansion step, perhaps in a somewhat simpler fashion, as the substitution of a pattern for a node, where the edges connecting to and from the pattern are already expanded. Hence, in order to bring our model closer to real logic, we may adopt the convention that the edges emanating from a node constitute a net. We call this set of edges the "output" of a node and we also specify that it expand identically. We must specify, for each expansion of a node, the following:

1. The graph g into which the node is expanded, its node count q, and its edge count n. We set q as a constant, and n is variable. We later derive a result for the average value of n.
2. The number s of sets into which the output net is expanded, and the nodes of the graph g to which the edges are connected.
3. The number of inputs generated from each input edge; we note that, because each input is the output of some other node, this is determined independently of this node. However, the way these edges are connected to the nodes of the graph g may be chosen at this point.

We now need to derive an expression for the number of substitutions which can be made at each node.

Given some parent graphs of M nodes, let us say that Step 1 can be performed in $\overline{R}^M$ ways, Step 2 in $\overline{S}^M$ ways, and Step 3 in $\overline{T}^M$ ways, so that the total number of ways is given by $(\overline{R}\ \overline{S}\ \overline{T})^M$. We compute $\overline{R}$, $\overline{S}$, and $\overline{T}$ on the basis of the descendant nodes being labeled. We assume that no isomorphism among the descendant nodes then requires us to divide by $q!$; this approximation was already discussed. Then

$$w^M = (\overline{R}\ \overline{S}\ \overline{T}/q!)^M. \tag{17}$$

We now derive, in turn, $\overline{R}$, $\bar{S}$, and $\overline{T}$. For a graph of q labeled nodes, there exist $q(q-1)$ possible ways of assigning a directed edge. n edges, when no multiple connections between pairs of nodes are permitted, can be assigned in $\binom{q(q-1)}{n}$ ways. If we specify an average value $\bar{n}$ for the number of edges internal to a graph for the substitutions, we can see that we satisfy essentially the conditions in Eq. (14) and can apply Eq. (15), that is,

$$\log_2 \overline{R} = \bar{n} \log_2 q(q-1)/\bar{n} + [q(q-1) - \bar{n}] \times \log_2 [q(q-1)/q(q-1) - \bar{n}]. \tag{18}$$

The selection of the expansion factors of the nets (Step 2) we take simply to be a selection of s blocks out of the q blocks of the graph g. There are

$$\binom{q}{s} \tag{19}$$

ways of selecting these. We also fix the average value $\bar{s}$ of s, so that we again satisfy Eq. (14) and can use again Eq. (14) to determine $\bar{S}$; i.e., we have

$$\log_2 \bar{S} = \bar{s} \log_2 q/\bar{s} + (q-\bar{s}) \log_2 q/(q-\bar{s}). \tag{20}$$

The factor $\overline{T}$ requires a somewhat different technique for evaluation. We note that we have in the unexpanded (i.e., the parent) graph $\bar{f}M$ edges, each of which generates a descendant edge set. The source (output) nodes of these edge sets and their sizes are selected in Step 2; in Step 3 we select the sink or input nodes. The number of ways this can be done is

$$(q)s = (q)(q-1)\cdots(q-s+1)$$

if the edge is split into s descendant edges. Let c_s be the fraction of edges which have s descendants. Then we find that

$$\overline{T}^M = \prod_s [(q)s]^{c_s \bar{f} M}. \tag{21}$$

Equations (11), (15), and (19) can be combined to give us

$$c_s = \binom{q}{s} \left(\frac{\bar{s}}{q-\bar{s}}\right)^s \left(\frac{q-\bar{s}}{q}\right)^q \tag{22}$$

and, substituting into (21), we find

$$\log_2 \overline{T} = \sum_s \bar{f} \binom{q}{s} \left(\frac{\bar{s}}{q-\bar{s}}\right)^s \left(\frac{q-\bar{s}}{q}\right)^q \log_2 (q)s. \tag{23}$$

We make an approximation in treating average values as if they were actual values. One can see that the error due to the approximation is of the order of $M^{1/2}$, i.e., M log $\overline{T}$ would be off by errors of about $M^{1/2}$, which is negligible. We find, then, for $\overline{T}$

Table 2 Information content per circuit of logic (bits/circuits)

q	$\bar{f}$	$\bar{s}$	$\log \bar{s}/\log q$ [*Rent exponent*]	$\bar{I}$
10	2	5	0.699	6.95
		4	.602	6.99
		3	.477	6.86
10	2.5	6	.778	8.36
		5	.699	8.54
		4	.602	8.50
		3	.477	8.24
10	3	5	.699	10.04
		4	.602	9.88
		3	.477	9.49
12	2.5	7	.783	8.76
		6	.721	8.92
		5	.648	8.91
		4	.558	8.77
		3	.442	8.49
20	2.5	9	.733	10.00
8	2.5	4	.667	8.08
7	2.5	3	.565	7.74
6	2.5	3	.613	7.46
5	2.5	3	.683	7.01

$$\log_2 \overline{T} = \sum_s \bar{f} \binom{q}{s} \left(\frac{\bar{s}}{q}\right)^s \left(\frac{q-\bar{s}}{q}\right)^{q-s} \log_2 (q)s. \tag{24}$$

Using Eq. (17) with

$$w = \overline{R}\ \overline{S}\ \overline{T}/q! \tag{25}$$

and Eqs. (18), (20), and (24) we can compute values of w for different values of $\bar{n}$, $\bar{f}$, $\bar{s}$ and q. $\bar{n}$ can be related to $\bar{f}$, $\bar{s}$, and q by the self-similarity principle, as follows.

The total number of edges in the parent graph is given by $\bar{f}M$, where the descendant graph has $(\bar{f}\,\bar{s} + \bar{n})\ M$ edges, with qM nodes. Because the average number of edges per node should not change, we find that $\bar{f} = (\bar{f}\bar{s} + \bar{n})\ M/Mq$ or that

$$\bar{n} = \bar{f}(q - \bar{s}). \tag{26}$$

It is now possible to compute $(\log_2 w)/q - 1$ using Rent exponent p, [see Eqs. (3) and (4)], average fan-in $\bar{f}$, and q, which are given in Table 2. The value 8.5 quoted in this paper is for $q = 10$, $p = 2/3$, and $\bar{f} = 2.5$; the latter two values are fairly typical of computer logic. From Table 1 we can see, furthermore, that $\bar{I}$ [i.e., $\log_2 w \div (q-1)$] is relatively insensitive to q and $\bar{p}$, and moderately sensitive to $\bar{f}$ (i.e., if $\bar{f} = 2$, the $\bar{I}$ would be 7). In terms of system design this result indicates that consideration of too many entities at one step of the design process is not helpful.

Delay considerations

We are concerned here with the evaluation of memory logic in terms of delay. We use the result that the aver-

age path length L in a combinational logic complex is given by a relationship [1, 2]

$$L = C^{\alpha}, \tag{27}$$

where L may be considered either as the average or maximum path length with slightly different values of α. We give a derivation of Eq. (27) in Appendix B. Crudely,

$$\alpha + p \approx 1. \tag{28}$$

In the empirical studies [2] α ranged from 0.36 for a high speed machine to 0.65, and $\alpha + p$ was, in 4 out of 5 cases, a value of 1.1. From the data in that report we can estimate α to be 0.43 for typical logic. In addition, we need to know the particular equivalence ratio R for a given type of memory array, the response time t_{memory} for that array, and its bit count M. For the case

$$t_{\text{block}} = t_{\text{memory}}/(M/R)^{\alpha},$$

in the absence of better information, one would use $\alpha = 0.45$.

Appendix A: Derivation of the terminal count relationship for the design process [1]

We have the relationship $T = AC^{p}$, where C is the number of blocks in the logic complex and A is the average number of connections per elemental block.

We show here that this relationship is a consequence of the design process. At some stage of the design process, we have M logic blocks in our graph, with an average number of terminals for each block of A. In the hierarchical expansion process, all the blocks descendant from one particular block do not generate any new terminal lines outside, but the current external connections or terminal of this one particular get replicated, on the average, by a factor $\bar{s}$ at each step of the expansion. We show first that, after l steps, the expected replication is $\bar{s}^{l}$.

Let $p(s)$ be the probability that an edge is replicated s times in one step. Then $\Sigma p(s) = 1$ and $\Sigma s p(s) = \bar{s}$.

The generating function $P(x)$ can be written [5]

$$P(x) = \sum_{s} p(s)\, x^{s}$$

and

$$P(1) = 1,$$

$$[dP(x)/dx]_{x=1} = \bar{s}.$$

Let us consider the probability $p_l(S)$ that after l stages of the design process this single edge has generated S edges, and the associated generating function

$$P_l(x) = \sum_{S} p_l(s)\, x^{S},$$

which has also the property that

$$P_l(1) = 1,$$

$$[dP_l(x)/dx]_{x=1} = \bar{S}_l. \tag{A1}$$

The following is true [7]:

$$P_{l+1}(x) = P_l[P(x)].$$

Proof

$$p_{l+1}(s) = \sum_{s_0} p_l(s_0)\ \Sigma\ \{s_j : \Sigma s_j = s\} \prod_{j=1}^{s_0} p(s_j),$$

$$P_{l+1}(x) = \sum_{s} p_{l+1}(s) x^{s} = \sum_{s} x^{s} \sum_{s_0} p_l(s_0)$$

$$\times\ \Sigma\ \{s_j : \Sigma s_j = s\} \prod_{j=1}^{s_0} p(s_j)$$

$$= \sum_{s} \sum_{s_0} p_l(s_0) \Sigma \{s_j : \Sigma s_j = s\} \prod_{j=1}^{s_0} p(s_j) x^{s_j}$$

$$= \sum_{s_0} p_l(s_0) \sum_{s} \Sigma \{s_j : \Sigma s_j = s\} \prod_{j=1}^{s_0} p(s_j) x^{s_j}.$$

Because

$$\sum_{s} \Sigma \{s_j : \Sigma s_j = s\} = \Sigma \{s_j\},$$

we have

$$P_{l+1}(x) = \sum_{s_0} p_l(s_0) \Sigma \{s_j\} \prod_{j=1}^{s_0} p(s_j) x^{s_j}$$

$$= \sum_{s_0} p_l(s_0) \prod_{j=1}^{s_0} \sum_{s_j} p(s_j) x^{s_j}$$

$$= \sum_{s_0} p_l(s_0)\ P(x)^{s_0}$$

$$= P_l[P(x)]. \qquad \text{Q.E.D.}$$

This leads to the result that

$$\bar{S}_l = [(dP_l(x)/dx)]x = 1$$

$$= \bar{s}^{l}. \tag{A2}$$

Proof. The above is true for $l = 1$.

$$dP_l(x)/dx = \{dP_{l-1}[P(x)]/dx\}x = 1$$

$$= \{dP_{l-1}[P(x)]/dP(x) \cdot dP(x)/dx\}x = 1$$

$$= [dP_{l-1}(P)/dP]x = 1 \cdot [dP(x)dx]x = 1$$

$$= \bar{S}_{l-1} \cdot \bar{s}.$$

Induction completes the proof. Q.E.D.

The entire statement

$$T = A\bar{s}^{l} \tag{A3}$$

is true because A and $\bar{s}^{l}$ are independently determined average quantities. Because

$$l = \log C / \log q, \tag{A4}$$

we can see that

$$T = A\bar{s}^{\log C/\log q}$$

$$= AC^{\log \bar{s}/\log q},$$

which allows us to relate $\bar{s}$ to the Rent exponent p by

$$p = \log \bar{s} / \log q. \tag{A5}$$

Appendix B: Average path length relationship

We wish to show that it is plausible that the expectation value of the number of blocks I on a path through a combinatorial logic with C circuits is given by

$$\bar{L} = C^{\alpha}. \tag{B1}$$

Let us specialize our design process for combinational circuits and consider the number of blocks from input to output as well as the number of paths from input to output. Let us also consider the set of graphs to be used for substitution in the design process; let the average lengths of paths through such graphs be $\bar{\lambda}$, and the average number of descendant paths for any input be given as $\bar{m}$. Then, assume that a path of length r exists in a complex, after one further step of the design process the expected length of this path is given as

$$r\bar{\lambda}, \tag{B2}$$

while the number of descendant paths is

$$r^{\bar{m}}. \tag{B3}$$

The average length of the paths after one design step is then greater than

$$\bar{r}\bar{\lambda}, \tag{B4}$$

where $\bar{r}$ is the average length of the parent paths. We then find

$$L(I) \geq \bar{\lambda}^{l}. \tag{B5}$$

Equation (B5) would be an equality if the substitution graphs would be of a "regular form," where the number of blocks in any path is λ and the number of blocks in the graph at any level of the path is s (e.g., Fig. 1 represents such a graph if we omit the input feeding the block at the upper right-hand corner). In such a special case we have

$$s\lambda = q,$$

$$\log s/\log q + \log \lambda/\log q = 1,$$

$$p + \alpha = 1. \tag{B6}$$

In fact, it was found [2] that $\alpha + p$ was about 1.1.

References

1. W. E. Donath, "Stochastic Model of the Computer Logic Design Process," IBM Research Report RC-3136, November, 1970, IBM Thomas J. Watson Research Center, Yorktown Heights, New York.
2. W. E. Donath and R. B. Hitchcock, "Path Lengths in Combinational Computer Logic Graphs," IBM Research Report RC-3383, June, 1971, IBM Thomas J. Watson Research Center, Yorktown Heights, New York.
3. A. Weinberger, private communication.
4. B. Mandelbrot; e.g., "The Pareto-Levy Law and the Distribution of Income," *International Econ. Rev.* **I**, 79 (1960); Information Theory and Psycholinguistics: A Theory of Word Frequencies," from *Readings in Mathematical Sciences*, P. F. Lazarsfeld and N. W. Henry, eds., MIT Press, Cambridge, Massachusetts, 1968, p. 350.
5. K. L. Chang, *Markov Chains with Stationary Transition Probabilities*, Springer Verlag, New York, 1967.
6. E. Rent, private communication.
7. B. S. Landman and R. L. Russo, "On a Pin-vs-block Relationship for Partitions of Logic Graphs," *IEEE Trans. Electronic Computers* **C-20**, 1469 (December, 1971).

Received November 12, 1973

The author is located at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598.