for each class, but in doing so it is necessary to use a procedure such as that of Fukunaga and Koontz [4] to ensure that for class C_i, the resulting feature vector has the maximum ability to separate class C_i from the mixture of the remaining classes.

EXPERIMENTAL RESULTS

In order to test the heuristics presented in this correspondence for attacking the problems of variable and dynamic dimensionality, a small experiment was performed using the A's, L's, and T's of Munson's data base. The data base contains one sample of each character written by 147 different writers. Each sample is a 24×24 binary matrix, and 5 features were measured by summing the following regions shown in Fig. 1:

$$x_1 = b + e + h$$

$$x_2 = c + f + i$$

$$x_3 = a + b + c$$

$$x_4 = d + e + f$$

$$x_5 = g + h + i.$$

The mean vectors are

$$\mu_A = (60,42,41,49,46)$$

$$\mu_L = (38,11,19,20,47)$$

$$\mu_T = (55,19,67,20,21),$$

and it can easily be seen that a feature like x_5 is good in discriminating between L and T but not in separating A from L. The following experiments were performed assuming normal class densities.

1) Bayes with equal *a priori* probabilities and five features.

2) e_i^2 and five features.

3) e_i^2 and five features. However, for all three classes feature 2 was deleted by our heuristic of setting x_2 equal to its mean.

4) e_i^2 and three features: $A - x_2,x_3,x_4$; $L - x_1,x_2,x_3$; $T - x_2,x_3,x_5$.

5) e_i^2 and χ^2. Five features for A and three features for L and T: $L - x_1,x_2,x_3$; $T - x_2,x_3,x_5$.

6) e_i^2 and χ^2. Five features for A, L and three features for T: $T - x_2,x_3,x_5$.

7) e_i^2 and χ^2. Five features, except for T, x_1 and x_4 were set equal to their means. (See Table I.)

Although the confusion matrices differed slightly, the error rate for the Bayes' classifier 1) was identical to that based on e_i^2. The remaining results reflect primarily the decrease in the number of features used. From the confusion matrices it is evident that most errors occurred by confusing A and L. Consequently we performed several experiments using more feature measurements on hypothesized A's and L's. In particular, notice that in 6) and 7) there is no difference between using the exact or approximated covariance matrix for T.

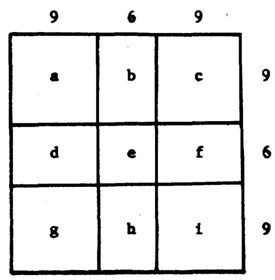

Fig. 1. Partitioning of data matrix.

TABLE I
EXPERIMENTAL RESULTS

Experiment	Error Rate (%)	Features
A	1.4	5
B	1.4	5
C	2.9	4
D	5.2	3
E	3.6	5-A; 3-L,T
F	2.0	5-A,L; 3-T
G	2.0	5-A,L; 3-T

CONCLUSIONS

Basing decisions in conventional as well as contextual recognition systems upon the statistics, $(x - \mu_i)^t\Sigma_i^{-1}(x - \mu_i)$, $1 \leq i \leq k$, is a computationally attractive alternative to using optimal techniques, particularly since one rarely knows enough about a problem to achieve optimality anyway. When computing only these quadratic forms, one also has the options of easily reducing dimensionality in any of the quadratic forms or of varying dimensionality dynamically by using p_i instead of the quadratic form. One achieves a degree of design independence in the recognition system, since instead of solving a single k-class problem k 2-class problems are really being solved.

In our experiments we have not observed any decrease in performance using our heuristics that cannot be explained by considering the number of feature measurements made.

REFERENCES

[1] J. K. Chang and S. J. Dwyer, "New multiclass classification method: Modified maximum likelihood decision rule," *Proc. 1st Int. Conf Pattern Recognition*, Washington, 1973, pp. 334–339.

[2] J. D. Patterson, "Linear mean square estimation technique of recognition," *Proc. 1st Int. Conf. Pattern Recognition*, Washington, 1973, pp. 67–72.

[3] H. O. Lancaster, *The Chi-Squared Distribution*. New York: Wiley, 1969.

[4] K. Fukunaga and W. L. G. Koontz, "Application of the Karhunen–Loève expansion to feature selection and ordering," *IEEE Trans. Comput.*, vol. C-19, pp. 311–318, Apr. 1970.

Matrix Transformations for N-Tuple Analysis of Binary Patterns

H. C. RATZ

Abstract—**The frequency of occurrence of logic functions of binary N-tuples can be observed from sequences of binary patterns. The logic functions considered here are AND, NOR, NAND, OR, and an odd-even parity check; and the frequency parameters are expressed as real matrix transformations on the probabilities of the patterns. Some properties, inverses, and interrelationships among the parameter sets are given, along with fast algorithms to facilitate computational processes. The results permit the outputs of convenient hardware logic operations to be converted into other parameters for smoothing, detection, or inference purposes, or to estimate the pattern probabilities by inversion. A measure of association among binary patterns is given as one characteristic feature which can be derived from the observed parameters.**

Manuscript received August 20, 1974; revised August 19, 1975.
The author is with the Department of Electrical Engineering, University of Waterloo, Waterloo, Ont., Canada.

***Index Terms*—Binary patterns, conditional expectation, fast algorithms, matrix transformations, N-tuple analysis, pattern association measure.**

I. Introduction

The analysis of blocks or patterns of n binary digits forms the basis for many studies in pattern recognition, uniform block coding, and the design of both algebraic and switching functions of binary variables [5], [12], [15], [21]. A sequence of such blocks may be observed from a source of patterns which are inputs to switching functions for analysis or recognition purposes, or as synchronized parallel channels of binary variables. The matrix transformations presented here are in ordinary real arithmetic and operate on a vector F of the probabilities of all 2^n possible binary patterns, rather than on the patterns themselves as in the case of transformations involving dyadic operations (see, for example, [9]). Hence, an analysis of the patterns can be made by observing the relative frequencies of all 2^n components of F; or, by directly observing functions of all the various N-tuples formed from the binary cells in the patterns. We discuss five such functions as matrix transformations of the vector F to give new 2^n-dimensional column vectors of observed parameters. Four of these parameter vectors represent logic operations on the binary pattern components while the fifth is a parity check on the N-tuples.

Each transformation is presented in recursively partitioned form, as are the inverses and the interrelationships among the parameter vectors where they exist. All nonsingular matrix transformations are factored into direct product matrices to provide fast algorithm computational methods. Computational complexity is important in image encoding [8] and some indication of the complexity of the arithmetic operations involved in these transformations is given.

As an application of these transformations, we introduce a measure of association among binary patterns which includes patterns that are not used with equal frequency.

II. Notational Representation

First we define a notation which permits the description of sets of binary patterns, including different probabilities of occurrence if required. The sequence of 2^n integers

$$k = 1, 2, 3, \cdots 2^n$$

enumerates the complete set of uniform blocks of n binary digits each, which can all be represented in an unique way by writing the number $(k-1)$ in the equivalent binary number form. A convenient notation is obtained by letting these ordered binary digits be the components of an n-dimensional vector. We define this vector as

$$\beta(k-1) = (b_n, b_{n-1}, \cdots, b_j, \cdots, b_1)$$

where

$$(k-1) = \sum_{j=1}^{n} 2^{j-1} b_j \leq 2^n - 1.$$

Thus, a value of the index k will be associated with that particular pattern obtained by writing $(k-1)$ in binary notation and then as the components of an n-dimensional vector, $\beta(k-1)$.

Next we define a 2^n-dimensional column vector

$$F = [f_k] = (f_1, f_2, \cdots, f_{2^n})^T$$

whose components are indexed by k and are each associated with the binary pattern given by $\beta(k-1)$. The values of the components f_k are the relative weights, frequencies of use, or probabilities of the corresponding binary block patterns. This notation enables transformations using real arithmetic on the probabilities of binary patterns to represent certain operations on the binary patterns themselves. The only restriction on the components of F is that they be normalized as probabilities.

We can treat, as a special case, a type of problem in which we are concerned only with the m different unique members of a set of binary patterns and not with their probabilities, by taking the f_k values for the members of that set as equal to $1/m$ and $f_k = 0$ for the $(2^n - m)$ others. In this way, the vector mF defines the set because its components mf_k are 1 or 0 according to whether the corresponding pattern $\beta(k-1)$ is a member of the set or not.

III. Transformations

The analysis of patterns of n binary cells may use features derived by classification schemes based on sets of N-tuples in the patterns. The total number of such N-tuple combinations is 2^n for N from 0 to n, and these can all be enumerated uniquely by the index

$$i = 1, 2, 3, \cdots, 2^n$$

where a particular N-tuple is defined by the positions of the 1's in $\beta(i-1)$. Then features or parameters may be derived from logic relationships applied to those particular cells of a pattern selected by an N-tuple. We consider four such logic relationships; namely, AND or "conjunction," NOR or "disjunction," NAND, and inclusive -OR.

In the conjunctive classification scheme, we obtain as observed parameters, the frequency of occurrence of the conjunction of those binary variables selected by the 1's in $\beta(i-1)$ over the set of patterns described by F. Such a scheme transforms the probabilities of the patterns given by F into 2^n parameters giving the probabilities of conjunction for all the possible N-tuples. This latter ordered set of parameters forms a 2^n-dimensional column vector

$$\Psi = [\psi_i].$$

The observation of Ψ from the source F can be expressed as a matrix transformation by

$$\Psi = U_n F$$

where U_n is a square matrix of order 2^n. It has the following simple recursive structure:

$$U_{n+1} = \begin{bmatrix} U_n & U_n \\ \mathbf{0} & U_n \end{bmatrix}, \qquad \text{with } U_0 = 1$$

and $\mathbf{0}$ is an $n \times n$ square array of 0's.

In the disjunctive classification scheme, we apply a NOR-gate operation to the cells selected by the 1's in $\beta(i-1)$, and again form a 2^n-component column vector of the parameters which result from taking the observed binary positions in all possible combinations. This scheme transforms the source patterns as defined by F and $\beta(k-1)$ into an observed vector of parameters

$$\Omega = [\omega_i].$$

In matrix form the transformation is

$$\Omega = N_n F$$

where N_n is a square matrix of order 2^n. It has the following simple recursive structure

$$N_{n+1} = \begin{bmatrix} N_n & N_n \\ N_n & \mathbf{0} \end{bmatrix}, \qquad \text{where } N_0 = +1.$$

Again, if we apply a NAND logic operation to each selected N-tuple, a different set of 2^n parameters is obtained. These result from a matrix transformation on F described by a square matrix A_n of order 2^n with the following recursive structure:

$$A_{n+1} = \begin{bmatrix} A_n & A_n \\ 1 & A_n \end{bmatrix}, \qquad \text{where } A_0 = 0$$

and 1 is a $n \times n$ square array of 1's.

Finally, the 2^n parameters resulting from employing an inclusive-OR logic operation on each N-tuple, are obtained from a matrix transformation on F by a square matrix V_n of order 2^n with recursive structure as follows:

$$V_{n+1} = \begin{bmatrix} V_n & V_n \\ V_n & 1 \end{bmatrix}, \qquad \text{where } V_0 = 0.$$

However, the A_n and V_n matrices are singular and will not be considered further since the complete pattern probability distributions cannot be recovered from the observed parameters in these cases.

IV. Relationships Among the Transformations

The conjunctive transformation is equivalent to a multivariable analysis if we regard the cells $[b_j]$ of the patterns as binary random variables. Then F is the joint probability distribution function of the joint event k; namely, that the pattern is given by $\beta(k-1)$. If the complete set of multivariate moments of all orders among the binary random variables are observed, then the 2^n components of F can be derived. There are n first-order moments and $\binom{n}{r}$ moments of the rth order, namely,

$$E(b_{j_1} b_{j_2} \cdots b_{j_q} \cdots b_{j_r}) = \psi_m^{(r)}$$

where m takes on the values given by

$$m = \sum_{q=1}^{r} 2^{j_q - 1} + 1 \text{ and } |\beta(m-1)|^2 = r.$$

Thus, the parameters resulting from the transformation U_n are the multivariate moments of all orders if the elements of the patterns are regarded as random binary variables.

The characteristic function of the joint probability distribution F is its multivariate Fourier transform, which is the expected value of the kernel

$$r_{ik} = \exp\{\sqrt{-1}\,[\pi\beta(i-1)] \cdot [\beta(k-1)]\}$$

where the exponent contains an inner vector product. In normalized form, this is [7]

$$E(r_{ik}) = \phi_i = \frac{1}{\sqrt{2^n}} \sum_{k=1}^{2^n} (-1)^{\beta(i-1)\cdot\beta(k-1)} f_k.$$

This is an orthonormal transformation with inverse

$$f_k = \frac{1}{\sqrt{2^n}} \sum_{i=1}^{2^n} (-1)^{\beta(i-1)\cdot\beta(k-1)} \phi_i$$

and

$$2^{-n} \le \Sigma \phi_i^2 = \Sigma f_k^2 \le 1.$$

Once again, the 2^n parameters $\{\phi_i\}$, can be taken as the components of a column vector Φ, and the above transformations written as

$$\Phi = R_n F \qquad \text{and} \qquad F = R_n \Phi$$

where the transformation matrix R_n is proportional to a Hadamard matrix [4] of order 2^n with the simple recursive structure

$$R_{n+1}^{-1} = R_{n+1} = \frac{1}{\sqrt{2}} \begin{bmatrix} R_n & R_n \\ R_n & -R_n \end{bmatrix}, \qquad \text{where } R_0 = 1.$$

The R_n-matrix is related to a matrix of the Walsh functions by a reordering of the rows [20]. A closer examination of the R_n transformation shows that the r_{ik} are selective parity functions, that is,

$$\begin{aligned} r_{ik} &= (-1)^{\beta(i-1)\cdot\beta(k-1)} \\ &= -1, \quad \text{if } \beta(i-1)\cdot\beta(k-1) \text{ is odd} \\ &= +1, \quad \text{if } \beta(i-1)\cdot\beta(k-1) \text{ is even.} \end{aligned}$$

Then the 2^n parameters $\{\phi_i\}$ are the average frequency of parity checks for all possible selections of the n binary cells in the pattern.

The elementary properties of the parameters follow directly from the definitions. For example the ranges are

$$0 \le f_k \le 1 \quad 0 \le \psi_i \le 1$$
$$0 \le 2^n \phi_i^2 \le 1 \quad 0 \le \omega_i \le 1.$$

In the case of purely random patterns, $f_k = 2^{-n}$ for all k, and then

$$\phi_m^{(r)} = 0 \text{ and } \psi_m^{(r)} = \omega_m^{(r)} = 2^{-r}.$$

The first-order parameters are the ones for which the index i satisfies

$$|\beta(i-1)|^2 = 1,$$

that is,

$$i = 2^{j-1} + 1, \qquad (j = 1, 2, 3, \cdots, n),$$

and are designated by the subscript s, so that

$$s = 2, 3, 5, 9, 17, \cdots$$

and for these we have

$$\sqrt{2^n}\,\phi_s = \omega_s - \psi_s \text{ and } \omega_s + \psi_s = 1.$$

The R_n transformation is orthonormal and self-inverse, but the N_n and U_n transformations representing logic operations are not orthogonal. However, they are nonsingular and their inverses have the following simple recursive structure:

$$U_{n+1}^{-1} = \begin{bmatrix} U_n^{-1} & -U_n^{-1} \\ \mathbf{0} & U_n^{-1} \end{bmatrix} \text{ and } N_{n+1}^{-1} = \begin{bmatrix} \mathbf{0} & N_n^{-1} \\ N_n^{-1} & -N_n^{-1} \end{bmatrix}$$

where $U_0^{-1} = N_0^{-1} = +1$.

Introducing three new square matrices of order 2^n, namely, T_n, D_n, and S_n, we can write the relationships between the parameter vectors as follows:

$$T_n \Psi = \Phi = D_n \Omega \qquad \text{and} \qquad \Psi = S_n \Omega$$

or, alternatively,

$$T_n U_n = R_n = D_n N_n \text{ and } U_n = S_n N_n;\ N_n = S_n U_n.$$

All the transformations T_n, D_n, and S_n and their inverses can be constructed recursively, and are summarized here without further derivation. In all cases, we have

$$S_0 = T_0 = D_0 = T_0^{-1} = D_0^{-1} = S_0^{-1} = 1.$$

Then

$$T_{n+1} = \frac{1}{\sqrt{2}} \begin{bmatrix} T_n & \mathbf{0} \\ T_n & -2T_n \end{bmatrix} \text{ and } T_{n+1}^{-1} = \frac{1}{\sqrt{2}} \begin{bmatrix} 2T_n^{-1} & \mathbf{0} \\ T_n^{-1} & -T_n \end{bmatrix},$$

$$D_{n+1} = \frac{1}{\sqrt{2}} \begin{bmatrix} D_n & \mathbf{0} \\ -D_n & 2D_n \end{bmatrix} \text{ and } D_{n+1}^{-1} = \frac{1}{\sqrt{2}} \begin{bmatrix} 2D_n^{-1} & \mathbf{0} \\ D_n^{-1} & D_n^{-1} \end{bmatrix},$$

and

$$S_{n+1}^{-1} = S_{n+1} = \begin{bmatrix} S_n & \mathbf{0} \\ S_n & -S_n \end{bmatrix} \text{ and } S_{n+1}^{-1} = \begin{bmatrix} S_n^{-1} & \mathbf{0} \\ S_n^{-1} & -S_n^{-1} \end{bmatrix}.$$

V. Implementation and Fast Algorithms

The transformations and their inverses are useful in investigating the relationships among the parameter vectors for a given input pattern set. It is to be noted that the inverses involve subtractions as well as additions while the direct transformations U_n and N_n involve addition only. Moreover, as n increases, the zeros predominate in these latter transformation matrices which contain 3^n ones. A simple measure of computational complexity is the number of operations by a two-input adder/subtractor to effect the transformation; i.e., one less than the number of nonzero elements in a row. Using this definition, the complexity of the transformations, U_n, N_n, U_n^{-1}, N_n^{-1} is $(3^n - 2^n)$. For the orthonormal R_n transformation, the corresponding measure is $(2^{2n} - 2^n)$. Thus, the U_n and N_n transformations have a computational advantage over the R_n transformation, since they require only a fraction of the number of operations. This fraction is an order of magnitude for $n = 9$, and decreases exponentially with increasing n. Nevertheless, in all cases, the amount of computation required for large n is prodigious using a one step transformation.

Because of the recursive structure of all the matrices, we can derive so-called "fast" algorithms based upon a theorem of Good [11] which we will restate here in a convenient notation. Let M_n be a nonsingular square matrix of order 2^n constructed recursively according to

$$M_{n+1} = \begin{bmatrix} aM_n & bM_n \\ cM_n & dM_n \end{bmatrix}, \qquad \text{where } M_0 = 1.$$

The matrix M_n is called the nth "direct power" of the matrix M_1. Except for A_n and V_n, all the transformations we use are of this general type and fast algorithms result from the ability to factor such $2^n \times 2^n$ matrices into the product of n identical factors. For this purpose we define $G_n(M_1)$ as the $2^n \times 2^n$ matrix constructed from the four elements of the 2×2 matrix M_1 as shown in the following examples:

$$G_1(M_1) = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = M_1; \; G_2(M_1) = \begin{bmatrix} a & b & 0 & 0 \\ 0 & 0 & a & b \\ c & d & 0 & 0 \\ 0 & 0 & c & d \end{bmatrix}$$

$$G_3(M_1) = \begin{bmatrix} a & b & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a & b & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a & b & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a & b \\ c & d & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c & d & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c & d & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c & d \end{bmatrix}.$$

Note that $G_n(M_1)$ has, at most, 2^{n+1} nonzero entries. Now the theorem of Good [11] states that a matrix constructed recursively as for M_n, may be factored into n factors $G_n(M_1)$. This factoring is the basis of fast algorithms since the 2^{2n} products in M_n are replaced by at most $n2^{n+1}$ products.

Each of the transformations U_1, N_1, U_1^{-1}, N_1^{-1} contains one zero, so that $G_n(U_1)$, for example, requires 2^{n-1} operations of the two-input add/subtract type. The factoring requires a cascade of n such matrix multiplications for $n2^{n-1}$ operations in all. Similarly, the orthonormal transformation factors into a cascade of n factors, each requiring 2^n operations, for a total of $n2^n$ in all. The computational operations required by the direct transformations for the case when $n = 3$ are illustrated diagrammatically in Fig. 1.

All the transformations T_n, D_n, and S_n and their inverses are direct product matrices. They can be constructed recursively, and can be implemented by fast algorithms through factoring into n identical factors for each of the type $G_n(M_1)$ and of order 2^n. The T_1, D_1, and S_1 matrices each contain one zero, so that a 2^n-order transformation will require $n2^{n-1}$ operations of the fast algorithm. Therefore, we can conclude that the same number of computational operations are required to obtain the Φ set of parameters using the R_n Hadamard transformation, as are required to obtain either the Ψ-AND or the Ω-NOR set of parameters and then to convert them to the Φ set.

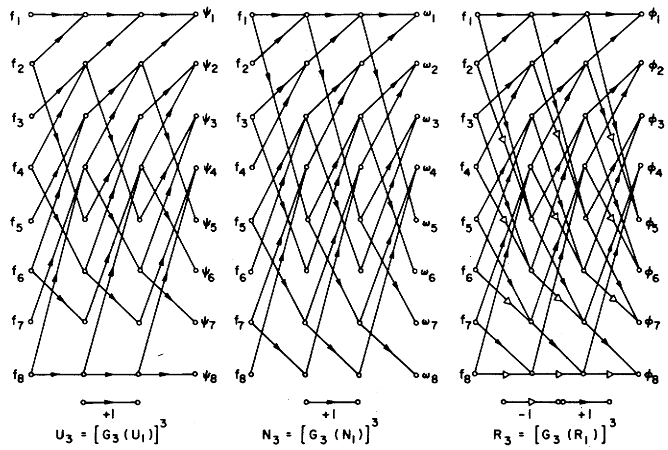

Fig. 1. Fast computational algorithms with $n = 3$.

VI. Applications

The application of these transformations in the analysis of patterns occurs when hardware considerations dictate that the observed parameters are to be the N-tuples of AND gates or NOR gates so that the U or N transformations are appropriate, or parity counters as with the R matrix [12], [13]. If we adopt the conjunctive classification scheme of the U-transformation, then we formulate the basis for the conditional probability computer described by Uttley [19] in which conditional probabilities are calculated from the ratio of two ψ-coefficients that differ by one in order.

The computation of conditional probabilities can form the basis for the design of a character identification system [6] and the frequency of logic conjunctions can be obtained directly with counters and logic gates. The R-transformation has the advantage of being orthogonal and leads to a conditional expectation computer using the ϕ coefficients as derived from parity counters on the N-tuples of the observed data [18].

Another example of the application of the parameters defined by these transformations is in the derivation of features or measures for pattern recognition purposes. For example, we may define a measure of the degree of association among patterns of a given set, or a measure of the distance between sets of patterns; and we may, in general, take different frequencies of occurrence into account. To do this, we indicate the first-order parameters with a subscript s, and the two sets of binary patterns defined by F_1 and F_2 vectors will have corresponding first-order parameters ϕ_{s1}, ϕ_{s2}, etc. Then the average Hamming distance between the two sets of binary patterns as given by [16] is

$$d = \frac{n}{2} - \frac{1}{2}\sum_s 2^n \phi_{s1}\phi_{s2}$$

$$= \sum_s [\omega_{s1} + \omega_{s2} - 2\omega_{s1}\omega_{s2}] = \sum_s [\psi_{s1} + \psi_{s2} - 2\psi_{s1}\psi_{s2}].$$

Now a measure of the degree of association A among one set of m patterns can be derived by setting $F_2 = F_1$ to obtain

$$A = \sum_s 2^n |\phi_s|^2 = n - 2d \le n - \log_2(m)$$

where $2d$ is descriptive of the "diameter" of the pattern space occupied by the set; that is, it is a measure of spread. The "effective volume v" occupied by the given set of patterns in terms of points in the total binary space is

$$m \le v = 2^{2d} = 2^n/2^A \le 2^n \text{ points.}$$

If the source consists of all possible patterns but they occur with unequal frequency then A will exceed zero and as the frequency of one pattern increases to predominate over all others the value of A increases towards n.

VII. Conclusions

The matrix transformations presented here can be employed in the design of networks for analyzing binary patterns [10], [12], [14]. The three matrix transformations R_n, U_n, and U_n are nonsingular, so that if the observations are complete (all N-tuples) then the probability distribution of the patterns (given by F) can be obtained with the inverse matrices. The N and U transformations describe logic operations for the processing of pattern data and involve fewer operations (about one-half, direct or inverse) giving a computational advantage over the orthogonal R transformation. The latter has been used for processing two-dimensional pictures [17] and for generalized spectral analysis [2]. Moreover, the AND and NOR operations can be implemented in hardware for fast convenient processing and the results subsequently converted to the overall equivalent of the orthogonal R transformation with no net increase in computational complexity. All the transformations and inverses that exist can be calculated using fast algorithm techniques. Thus, different types of smoothing can be obtained from the different transformations by removing the higher order parameters and then applying the inverse transformations. For example, smoothing of binary pattern images occurs when higher order parameters in the orthonormal expansion are neglected [1]; but, this can now be translated into a form that is applicable when NOR gates are used for data acquisition. As an example of a quantity, descriptive of observed patterns, a measure of association has been defined for a set of binary patterns which can be derived from either the NOR, AND, or parity parameters.

References

[1] N. A. Alexandridis and A. Klinger, "Real-time Walsh-Hadamard transformation," *IEEE Trans. Comput.* (Short Notes), vol. C-21, pp. 288–292, Mar. 1972.
[2] H. C. Andrews and K. L. Caspari, "A generalized technique for spectral analysis," *IEEE Trans. Comput.*, vol. C-19, pp. 16–25, 1970.
[3] H. Butin, "A compact definition of Walsh functions," *IEEE Trans. Comput.*, vol. C-21, pp. 590–592, 1972.
[4] A. T. Butson, "Generalized Hadamard matrices," *Proc. Amer. Math. Soc.*, vol. 13, pp. 894–898, 1962.
[5] P. Galingaert, "Switching function canonical forms based on commutative and associative binary operations," *AIEE Trans. Commun. Electron.*, vol. 79, pp. 808–814, 1961.
[6] C. K. Chow, "An optimum character recognition system using decision functions," *IRE Trans. Electron. Comput.*, vol. EC-6, pp. 247–254, 1957.
[7] R. P. Coleman, "Orthogonal functions for the logical design of switching circuits," *IRE Trans. Electron. Comput.*, vol. EC-10, pp. 379–382, 1961.
[8] B. J. Fino and V. R. Algazi, "Slant Haar transform," *Proc. IEEE* (Lett.), vol. 62, pp. 653–654, May 1974.
[9] L. T. Fisher, "Unateness properties of AND-EXCLUSIVE-OR logic circuits," *IEEE Trans. Comput.*, vol. C-23, pp. 166–172, Feb. 1974.
[10] K. Fukunga and T. Itô, "A design theory of recognition functions in self-organizing systems," *IEEE Trans. Electron. Comput.*, vol. EC-14, pp. 44–52, 1965.
[11] I. J. Good, "The interactive algorithm and practical Fourier analysis," *J. Roy. Statist. Soc.*, vol. 20B, pp. 361–372, 1958.
[12] E. E. Gose, "An adaptive network for producting real functions of binary inputs," *Inform. Contr.*, vol. 8, pp. 111–123, 1965.
[13] ——, "A synthesis technique for networks consisting of logical functions feeding a linear summation element," *IEEE Trans. Electron. Comput.*, vol. EC-14, pp. 254–256, 1965.
[14] T. Itô, "A note on a general expansion of functions of binary variables," *Inform. Contr.*, vol. 12, pp. 206–211, 1970.
[15] R. J. Lechner, "Transformations among switching function canonical forms," *IEEE Trans. Electron. Comput.*, vol. EC-12, pp. 129–130, 1963.
[16] J. Pearl, "Application of Walsh transform to statistical analysis," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-1, pp. 111–119, 1971.
[17] W. K. Pratt, J. Kane, and H. C. Andrews, "Hadamard transform image coding," *Proc. IEEE*, vol. 57, pp. 58–68, 1971.
[18] H. C. Ratz, "A new computer for process optimization," *Int. J. Contr.*, vol. 1, pp. 81–99, 1965.
[19] A. M. Uttley, "The design of conditional probability computers," *Inform. Contr.*, vol. 2, pp. 1–24, 1959.
[20] J. L. Walsh, "A closed set of normal orthogonal functions," *Amer. J. Math.*, vol. 45, pp. 5–24, 1923.
[21] H. C. Wehrfritz, "Techniques for the transformation of logic equations," *IEEE Trans. Comput.*, vol. C-23, pp. 477–480, May 1974.

A Note on Atrubin's Real-Time Iterative Multiplier

LAKSHMI N. GOYAL

***Abstract*—This correspondence presents a new multiplication algorithm for Atrubin's one-dimensional real-time iterative multiplier such that all the cells including the first cell in the array are identical in all respects, for the no-delay case.**

***Index Terms*—Arithmetic, iterative array, multiplication, on-line multiplier, real-time multiplier.**

I. Introduction

In his paper "A one-dimensional real-time iterative multiplier," [1] Atrubin presented the design of a real-time multiplier. His multiplier consists of a one-dimensional iterative bilateral array of cells (finite state machines) such that when the digits of two integers are presented to the cell at the extreme left end of the array a pair at a time, the same cell indicates the product digits at the rate of one per cycle. He had further shown that for the no-delay case the extreme left-hand cell or the initial cell had to be different from the rest of the cells which were all identical finite state machines in terms of the memory space needed as well as the control. In this correspondence we shall show that a no-delay real-time multiplication can be achieved by an identical set of cells including the initial cell, without increasing the complexity of the individual cell.

II. Structure of the Multiplier

For the sake of simplicity, we shall use, wherever possible, the notation used by Atrubin.

Suppose the two integers to be multiplied are

$$A = \sum_{l=0}^{n} a(i)2^i, \qquad a(i) \in \{1,0\}$$

$$B = \sum_{l=0}^{m} b(i)\,2^i, \qquad b(i) \in \{1,0\}$$

where m and n are arbitrary.

Let the product be given by

Manuscript received July 17, 1974; revised January 31, 1975. This work was supported in part by the National Science Foundation under Grant US-NSF-GJ-38204.

The author is with the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801.