taining these integrating elements. They are forced to introduce an arbitrary "stability margin" to allow for this and "leave it up to the user to determine what is required for his problem" even though their original objective had been to "get away from the unsatisfactory situation where simulation languages provide a number of integration formulas, from which the user can choose, but without having any criteria to guide his choice." Using the approach described in [1] and [2], on the other hand (which leads to the same stability analysis as has been given by Gray (3) and numerous others), readily gives the desired condition. For the class of formulas considered by Giloi and Grebe, the condition for stability of the whole system as distinct from just the integrating elements becomes that the roots of each of the polynomials \( D(z) - \lambda_i T(z) \) lie within the unit circle. (Here, \( C(z) \) denotes the numerator polynomial in their equation (7), \( T \) is the step length, and \( \lambda_i \) \((i = 1 \cdots n)\) are the poles of the system being simulated.)

It is a little surprising that, having adopted an analog, block-oriented viewpoint, the authors have not made use of the concept of total dynamic error which is now generally used to describe the imperfections in frequency response of analog computing elements. Instead they use the magnitude and phase errors which are the separate in-phase and quadrature components of total dynamic error. It is now generally realized that it is usually only the total dynamic error that counts as neither of the separate components is more serious than the other, except in special cases (such as when the solution to a second order system is required to look right to the eye). Giloi and Grebe, however, regard the phase error as being more serious than the magnitude error. They are therefore led to search for formulas having zero phase error and thence to prefer predictors of even and correctors of odd order.

The authors' general method of attack is to be commended in spite of the shortcomings that have been mentioned and the article should prove of interest to all those concerned with the digital simulation of continuous systems.

P. R. BENYON
Commonwealth Scientific and Industrial Research Organization
Australia

REFERENCES


This paper is an interesting probabilistic study of the decomposability of sequential machines. State-splitting is not admitted, however. In view of the well known relationship between decompositions (of the state behavior) of sequential machines and partitions having the substitution property (SP partitions [1]), the author’s interest turns to SP partitions.

Theorem 2.1 gives an accurate formula for \( E(n, p) \) the average number of nontrivial SP partitions of an \( n \)-state, \( p \)-input machine. This expectation formula is easily derived, but its computational complexity increases rapidly with \( n \). Hence the author’s decision to investigate the asymptotic behavior of the expectation function in question. It turns out that by applying rather involved bounding techniques, the author finally obtains surprisingly simple results. Namely, for \( n \to \infty \) we have \( E(n, p) \to \infty \), provided \( p \) increases as a function of \( n \) in such a way that \( p(n) \ln n \to n - \infty \) for \( n \to \infty \) (Theorem 2.4).

Conversely, \( E(n, p) \to 0 \) for \( n \to \infty \), provided \( p \) increases as a function of \( n \) such that \( p(n) \ln p(n) \to 0 \) for \( n \to \infty \) (Theorems 4.1).

The statistically minded reader will find interest in both the techniques and the final results of this paper. On the other hand, it is rather doubtful whether any practical conclusions as to efficient decomposition techniques can be drawn from the results of this paper. A relevant point is the fact that many SP partitions lead to very inefficient decompositions. Also, frequently only state-splitting will lead to useful decompositions. In this connection, this reviewer would like to state his viewpoint that it is usually good strategy to look immediately for useful preserved covers rather than searching for SP partitions first. For further details see [2], [3].

M. Yoel
Technion
Haifa, Israel

B. SWITCHING AND AUTOMATA THEORY


This paper concerns itself with rooted trees which have labeled nodes. The labels are taken from a stratified alphabet (each label is associated with a nonnegative number, the number of branches descending from it). The alphabet is divided into terminal and nonterminal labels. Tree generating grammars called regular systems are introduced. A set of trees serve as "axioms" and production rules of the form \( \Phi \to \Psi \), where \( \Phi \) and \( \Psi \) are trees, allow successive replacement of subtrees \( \Phi \) by \( \Psi \). The "language" generated by such a system is the set of trees generable from the axioms containing terminal labels only. Such languages, when trees are written linearly (prefix or postfix form) are context free.

When the grammars are written linearly, they are semi-Thue systems, since production rules may be length decreasing. However, the restriction that all strings must map into trees (there must be some stratification assignment) assures that the language generated is context free.

Tree automata are tree analogs of finite state machines. A tree automaton acts on the set of trees over some stratified alphabet, accepting some subset. Intuitively, such an automaton begins at the sequence of endpoints, assigning a state to each endpoint. Moving up the tree, it assigns a state to each node as a function of its label and the sequence of states assigned to the nodes directly below it. If the state assigned to the root node is a designated (final) state, the tree is accepted. A tree automaton may be deterministic or nondeterministic, but for each nondeterministic tree automaton there is a deterministic one accepting the same set.

The author proves that the set of languages accepted by the tree automata is exactly the set generated by the regular systems. Further, the nonterminal labels are not required for generating this set.

All alphabets, trees, sets of production rules, and sets of automata states discussed are finite. The theorems, in general, require these finiteness limitations in their proofs.

The definitions, theorems, and proofs are concise and precise. The casual reader is warned, however, that the abundance of formalism, the sparsity of examples, and the lack of motivating discussion and intuitive explication will probably make for difficult reading for those not familiar with the subject matter.

This paper is essentially a collection of some of the major theorems from the thesis of the author. A review of the basis (taken from the author’s abstract) appears in Computing Reviews, vol. 9, no. 2, February 1969. The review states the major theorems, and all but the last (on state minimization of tree automata) appear in this paper.

S. Bergman
Temple University


Stochastic (or, indifferently, probabilistic) sequential machines have been recognized to be important mathematical models for several classes of information processing systems with memory, in which random disturbances cannot be neglected. As examples it suffices to mention the
models of finite-state discrete channels for information transmission, sequential machines constructed from unreliable components, and adaptive learning systems. From a more abstract and less engineering-motivated angle, the stochastic sequential machine (SSM) represents the natural generalization of the deterministic one, on which a considerable body of knowledge exists. Not to be overlooked, finally, is the fact that the rich underlying structure promises elegant results and this undeniably aesthetic dimension certainly contributes an additional interest to this per se attractive topic.

A verification of the last statement is provided by this paper by C. L. Liu, in which the specific class of definite SSM’s is further investigated (previous important results on such SSM’s were obtained by among others, Paz [1]). To bring the reader into proper focus, we briefly recall that the next-state function of a finite-state SSM is described by a set \( \mathcal{P} \) of matrices of the same order, such that \( P^{n+1} \in \mathcal{P} \) is the stochastic post-multiplicative transition matrix corresponding to input state \( i \). The SSM is definite of order \( k \) if the product of any \( j \geq k \) matrices \( P \) is a matrix with identical rows. The importance of such class of SSM’s is recognized by considering, for example, its identity with the model of a discrete finite-memory channel, that is, a channel exhibiting the property that the distribution of the channel states after an input sequence of length \( j \geq k \) depends only upon the last \( k \) symbols and not on previous symbols or states (see, e.g., [2 ch. 7]). The connection with this class of channels is illuminating on the significance of the term “definite,” for its usual connotation of absence of feedback.

The investigation of the transition behavior of SSM’s presented in this paper aims at establishing necessary and sufficient conditions for an SSM in order to be definite of order \( k \). The analysis takes the very natural development of proceeding from the autonomous case (i.e., \( \mathcal{P} \) contains only one matrix \( P_i \) to the general input-dependent case. The basic formal tool adopted is a matrix reduction technique resulting in a sequence of stochastic matrices \( P, P_1, P_2, \ldots \). It is worth commenting that the reduction procedure lends itself to an interesting interpretation. We may think of \( P \) in a twofold manner: as a set of points \( Q^1, \ldots, Q^m \) of the \( n \)-dimensional simplex \( S \) represented by the rows of the matrix, and as of a linear transformation of \( S \) into itself (since \( P \) is stochastic). If \( P \) is simple (as required by definiteness), the transformation is many-to-one into, i.e., for all \( j \) each \( Q^j \) has a “projection” \( Q^i \) in a fixed simplex of minimum dimensionality so that \( Q^i \) and \( Q^j \) have the same image under \( P \). The reduction of Liu performs exactly this projection. Moreover, in a recursive manner, one can obtain \( Q^i \) so that \( Q^i \) and \( Q^j \) have the same image under \( P^k \). Clearly, then, a test for definiteness of order \( k \) (\( Q^i = Q^j \) for each \( i \) and \( j \)) is that \( k \) is the smallest index for which \( P_k \) is \( 1 \times 1 \).

After providing the necessary background with the autonomous case, the analysis is extended to input-dependent SSM’s. In this case the key role is played by a simultaneous reduction of all matrices \( P^k \) so that for all of them the projection occurs on the same simplex of minimum dimensionality. This procedure is then recursively performed on the set of reduced stochastic matrices \( P_1^1, P_1^2, \ldots \), and so on. The simultaneous reduction allows the important result that the projections of the points (rows) of the product of any \( k \) matrices of \( P \) lie on the same simplex as those of \( P^k \) and yields a generalization procedure for \( k \)th order definiteness, i.e., that the consistent reduced matrices \( P_1^1, P_2^1, \ldots \), (in the sense of the simultaneous reduction) be \( 1 \times 1 \).

As pointed out by the author, the main results—in the form of conditions for definiteness—are of a sufficient nature: a necessary and sufficient condition is obtained, and rather laboriously, only for 3-state SSM’s.

The paper is well written and clear (a couple of misprints are quite tolerable), a very refreshing experience when so much literature obeyes the vogue of unnecessary difficulty. The background review, the logical development, and the straightforward exposition make reading quite easy and pleasant. The significance of the paper lies both in its interesting content and in the simulation it offers with several unsolved problems, especially with regard to a tighter characterization of definite SSM’s. For all of these reasons it constitutes a recommendable reading for anyone interested in discrete systems.

Franco P. Preparata
Istituto di Elaborazione dell’Informazione
Pisa, Italy

References


This paper is concerned with techniques for the synthesis of binary output synchronous sequential machines. In particular, the proposed techniques can be used to realize such a machine as an interconnection of a finite set of identical two-component machines or modules. Furthermore, the module can be selected in advance since it depends only on the number of inputs and not on the structure of the machine to be realized.

Research on this type of synthesis is, of course, motivated by recent developments in batch fabrication techniques. Since it is now possible to produce a large number of interconnected components on a single monolithic substrate, the question of which circuits should be manufactured naturally arises. Modules of the type studied in this paper have been proposed as possible candidates since any one module can be used to realize many different sequential machines.

A module is said to be universal for a class of sequential machines if any machine in the class can be realized as a network constructed from a finite number of copies of this module. Weiner and Hopcroft [1] and Newborn [2] have shown that there exist universal modules for the class of all sequential machines for which the number of inputs does not exceed a given number. Although their modules are simple, the resulting networks may be quite complex. For example, the module used for the class of all binary input machines consists of only three two-input gates and a delay element, but the number of modules needed to synthesize an \( n \)-state machine could be as high as \( 2^n - 2 \). The modules and synthesis techniques introduced in this paper are generalizations of those of Weiner and Hopcroft [1] and Newborn [2], for which this bound on the number of modules is significantly reduced.

The authors treat the synthesis of binary input machines in detail and indicate how their methods can be easily extended to synthesize machines with more than two input symbols. They define a class of modules of varying complexity such that each module is universal for the class of all binary input machines. In fact, for each \( r \geq 1 \), there is a module \( M_r \) which has \( 2r + 1 \) input terminals and is composed of two \( r \)-input gates, three two-input gates, and a delay element. They then present two different synthesis techniques which may be used with any one of these modules.

All of these techniques are based on the principle that each of the modules in a network represents a subset of states of the machine it is to realize. Thus, the number of modules is bounded by the number of subsets of this machine’s state set. The purpose of this paper is to show that if one of the modules \( M_r \), for \( r \geq 2 \), is used, then it is not necessary to represent all of the possible subsets.

The first technique is straightforward and leads to networks, using module \( M_r \), which never need more than \( 2^{nr} \) modules to realize an \( n \)-state machine. The second technique is considerably less efficient, but if \( r \geq 2 \), it yields networks for which the number of copies of module \( M_r \) needed to realize an \( n \)-state machine is bounded, to within a constant factor, by \( n^{r+1} \log_2 n \).

In conclusion, the authors point out that these bounds do not appear to be very tight. They further suggest that in practice it should be possible to do much better than the bounds would indicate.

J. R. Jump
Rice University
Houston, Tex.

References