A Syntactic Methodology for Automatic Diagnosis by Analysis of Continuous Time Measurements Using Hierarchical Signal Representations

M. Borahan Tümer, *Member, IEEE*, Lee A. Belfore, II, *Senior Member, IEEE*, and Kristina M. Ropella, *Senior Member, IEEE*

***Abstract*—In this paper, we present a methodology for automatic diagnosis of systems characterized by continuous signals. For each condition considered, the methodology requires the development of an alphabet of signal primitives, and a set of hierarchical fuzzy automatons (HFAs). Each alphabet is adaptively obtained by training an ART2 AU: PLEASE DEFINE ART2. architecture with signal segments from a particular condition. Then, the original signal is transformed into a string of vectors of primitives, where each vector of primitives replaces a signal segment in the original signal. The string, in turn, is presented to the HFA characterizing that particular condition. Each set of HFA consists of a main automaton identifying the entire signal, and several sub-automata each identifying a particular significant structure in the signal. A transition in the main automaton occurs (i.e., the main automaton moves from one state to another) if the corresponding subautomaton recognizes a token where a token is a portion of the string of vectors of signal primitives with a significant structure. The fuzziness in automaton operation adds flexibility to the operation of the automaton, enabling the processing of imperfect input, allowing for toleration measurement noise and other ambiguities. The methodology is applied to the problem of automatic electrocardiogram diagnosis.**

***Index Terms*—AUTHOR, PLEASE SUPPLY YOUR OWN KEYWORDS OR SEND A BLANK E-MAIL TO keywords@ieee.org to receive a list of suggested keywords.**

I. Introduction

Automatic diagnosis of complex nonlinear systems is a challenging research area in science and engineering and plays an important role in both medical and industrial applications. The diagnostic accuracy provided by the diagnostic system is among the most important criteria for measuring the performance of the diagnostic system; yet a highly desirable feature is the applicability of the diagnostic system to a variety of systems. Complicating the problem is that signals from which the diagnosis is derived can be perturbed by noise, localized baseline wander, and measurement nonlinearities.

Two different approaches to automatic diagnosis have received attention. The first approach is that of the knowledge-based diagnostic tools [1]–[3]. A diagnostic tool based on this approach has a knowledge base and a rule base. The diagnostic tool is presented to the symptoms of the system under analysis. The tool checks the knowledge base to see what symptoms the presented input corresponds in the knowledge base. The tool then determines, using the rule base, which fault, or faults, the given symptoms characterize. This approach needs a large knowledge and rule base to achieve automatic diagnosis.

The other approach to automatic diagnosis consists of the analysis of signals that the system under analysis generates. This approach is based on pattern recognition [4]–[21]. The signals used in the analysis are usually in the time domain. Diagnostic tools that attempt automatic diagnosis using the latter approach can be classified in two groups: 1) tools that use the decision-theoretic approach, and 2) those that employ syntactic methods. Automatic diagnosis using the decision-theoretic approach is based upon extracting features from signals that bear the characteristics of a fault. Diagnostic tools that follow the decision-theoretic approach frequently employ adaptive signal processing techniques, such as wavelets [22], or neural networks [4]–[7], [9], [10], claiming that classical signal processing [23], [24] techniques are insufficient to deal with the imperfect and imprecise information obtained from the signal.

The decision-theoretic approach does not utilize syntactic information extracted from the structure of signals which can be essential for automatic diagnosis. Considerable research, making use of syntactic information important to automatic diagnosis has been conducted [11]–[21].

The syntactic approach considers signals as a sequence of tokens that contain important structural information. In the syntactic approach, the signal is decomposed into windows of samples (the signal is a sequence of samples, usually time samples). The decomposition is performed so that the windows of samples are small enough not to contain any structural information. These windows of samples are also called primitives. The original signal is transformed into a sequence of these primitives (since tokens have structures, and primitives are small enough not to have structures, each token is composed of a subsequence

Manuscript received June 17, 2000; revised February 17, 2002. A preliminary version of this work with an emphasis on the theoretical aspects was presented in the IEEE International Conference on SMC 3 in San Diego, CA in 1998. Another preliminary version focusing on an ECG application was presented in North America Fuzzy Information Processing Systems Conference in Pensacola, FL, in 1998. This paper was recommended by Associate Editor A. Kandel.

M. B. Tümer is with the Department of Computer Engineering, Marmara University, Göztepe 81040, İstanbul, Turkey (e-mail: bora@eng.marmara.edu.tr).

L. A. Belfore, II, is with the Department of Electrical and Computer Engineering, Old Dominion University, 231D Kaufman Hall, Norfolk, VA 23529 (e-mail: before@ece.odu.edu).

K. M. Ropella is with the Department of Biomedical Engineering, Marquette University, Milwaukee, WI 53201 (e-mail: k.ropella@marquette.edu).

Digital Object Identifier 10.1109/TSMCB.2002.804365

of primitives). The transformed signal is then syntactically analyzed to determine if the transformed signal characterizes any fault in the system under analysis.

Different syntactic methods have been proposed for automatic diagnosis [11]–[21]. While some of the syntactic methods use regular state machines to perform syntax analysis [11], [16], [17], [19], [21], others involve more complex state machines, such as attributed and fuzzy state machines designed to deal with imperfect and imprecise information [8], [12]–[15], [18], [20]. References [8], [12]–[15], [18] use attributed state machines to make regular state machines more robust and reference employs fuzzy state machines.

Attributed state machines, functioning by formal grammars are more powerful than classical state machines, since classical state machines cannot handle the complexity of signals generated by nonlinear systems [13]. However, attributed methods require a great number of parameters to control the parsing process of the signals [13]. Studies based on attributed automata [16], [17], [19], [21] have shown that extensive computation is necessary to handle these parameters, and hence, attributed state machines have practical limitations [13]. Moreover, more robust systems are needed to deal with the noisy, subject-varying, and time-varying signals [13].

In this paper, we present a diagnostic tool for systems characterized by continuous signals. To handle various signal disturbances mentioned in the above paragraph, we use hierarchical fuzzy automatons (HFA) for the recognition of signals to achieve automatic diagnosis. HFAs are fuzzy automatons [25]–[28] that process a signal at several levels of detail. Moving up each level in the hierarchy results in the identification of more complex and global structures. At the apex of the hierarchy, there is one fuzzy automaton that recognizes a string representative of a condition. The input to the HFA is the time sampled signal that has been tokenized into primitives using an adaptive resonance theory 2 (ART2) artificial neural network (ANN) [29] where fuzziness of primitives has been extracted in an ad hoc fashion from the internal state of ART2. Nondeterministic operation of individual HFAs is an essential feature of its operation. The nondeterministic fuzzy automaton supports simultaneous transitions from any starting state to all potential next states. As the state machine operates, memberships within all states evolve until the state memberships along the transition paths dominate. As these states are identified, the HFA state memberships collapse into a small number of states for any given transition. Once this synchronization is achieved, the diagnosis is determined by examining the respective performance of several HFAs. An HFA is associated either with each different condition or with significant variations of the same condition. Thus, several HFAs are operated simultaneously on the same signal. The condition associated with the HFA can be associated with the highest membership that indicates the identified condition or by some other metric, such as the number of transitions that were matched with signal segments. The contributions of this work are three-fold.

1) An artificial neural system was used to produce the set of primitives (i.e., primitive alphabet). By doing this, we complemented the syntactic approach with the decision-theoretic approach (i.e., we adaptively constructed the primitive alphabet). Other syntactic methods use primitives to transform signals regardless of what fault is considered. On the other hand, the adaptive construction of the primitive alphabet in this study provides that each fault has its own alphabet (the neural system is trained for each fault by signals generated by a system that has this fault and learns primitives from these signals). This way, it is more likely that a signal that characterizes a specific fault can be more effective (with a smaller error) transformed by the primitive alphabet corresponding to that fault than by those of other faults.
2) The HFAs used in this study employ a two-folded fuzziness (the state fuzziness and the transition fuzziness), that increases the robustness of the fuzzy state machines over state machines used in existing methods. By transition fuzziness, a state machine is given the flexibility to make multiple transitions simultaneously. The state fuzziness provides a state machine with the capability of being at multiple states at the same time.
3) With the input synchronization capability of the presented diagnostic system, we can analyze signals regardless of the point at which the user of the diagnostic system starts to present the input signal to the system. This saves the user from having to make a priori modifications on the original input signal to present the signal starting from a predetermined point.

This paper is organized into eight sections including an introduction, a diagnostic system overview and operation, a presentation of the methodology for constructing the diagnostic system,

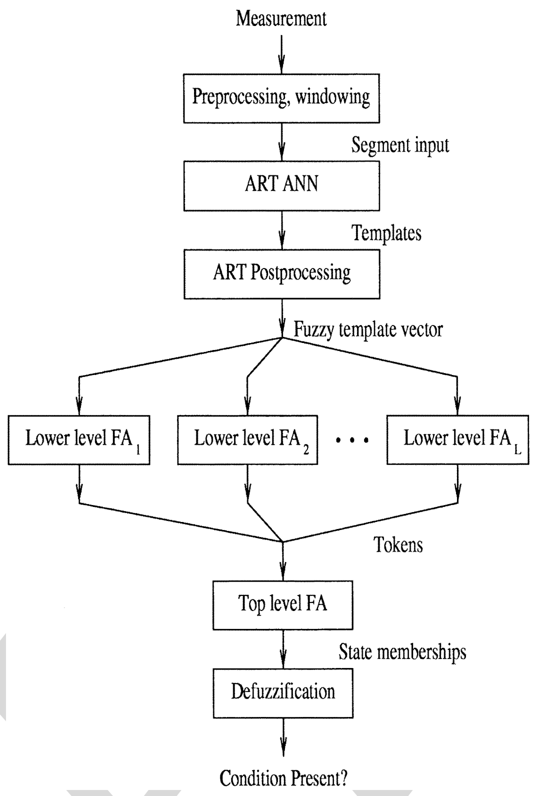


Fig. 1. Overview of diagnosis system processing for one condition.

```
Preprocessing
  • Depending on how noisy the signal is, several filtering methods may be employed.
  • As an example, for wandering baselines, a repeated moving average filter may be used or,
  • a low-pass filter may be used for high frequency noise involved.

Windowing
  • assume window size $w = k$ time samples
  • In training mode:
        shift step $s = l$ time samples where $l < k$
        do for each signal in the training set
            take the next $k$ samples to form segment $i$
            slide window $l$ samples forward
        until all samples are used.
  • In operation mode
        shift step $s = k$ time samples (i.e., $w = s$)
        do for the signal under analysis
            take the next $k$ samples to form segment $i$
            slide window $l = k$ samples forward
        until all samples are used.
```

Fig. 2. Pseudocode for preprocessing and windowing.

a discussion of HFAs, an application of automatic ECG diagnosis, the results of the analysis on the example, a conclusion, and a summary.

II. System Overview and Operation

In our approach, an HFA is constructed for performing automatic diagnosis. A block diagram of the system is shown in Fig. 1. Each box in Fig. 1 is illustrated with further details in Figs. 2–4. The diagnostic system is composed of one such subsystem, as in Fig. 1, for each condition. An ART2 ANN is used to tokenize the input signal for processing by the automatons that follow. The reason for using an ANN to tokenize the input signal is the need to adaptively construct a set of signal primitives from a group of signals characterizing a specific condition or fault. By this adaptive construction, the set of signal primitives is generated in a system-independent fashion. We call these signal primitives *templates*. The set of templates is called an *alphabet*. While ANNs prove necessary for the adaptive extraction of the templates, the decision-theoretic nature of ANNs imply their insufficiency for automatic diagnosis by signals with complex temporal structures. Automatic diagnosis based only on ANNs either fails to utilize the temporal structures of the signal under analysis, or attempts to express the structures as features of the signal, which becomes highly inefficient. The benefit of using the ART2 architecture is that learning is unsupervised, and has the ability to identify shapes in segments of the input signal. Furthermore, for each input segment, ART2 can also be modified to supply a measure which can be used as a fuzziness measure for membership to each ART2 class. The measures are passed, as a vector, to the sub-automata. Each subautomaton recognizes a syntactic structure (token) from the input measurement. When a subautomaton reaches an accepting state, a transition results in the main automaton. In this work, the HFA acts as a nondeterministic finite automaton, with simultaneous membership in several states and transitions along several paths possible. Diagnosis for a particular condition (i.e., the system either rejects the existence of that condition, or else indicates with a particular certainty, that the input signal characterizes that condition) is achieved, either when all input segments are processed by the corresponding HFA, or when the HFA fails to move at a certain input segment. Each condition for which diagnosis is desired requires the development of its own HFA system and training of the ART2 ANN. The full details of the design and operation of the system are presented in [30].

A. Operation of the HFA

A key aspect of the diagnosis system is the operation of the HFA. Each HFA consists of a single main automaton with transitions driven by a set of sub-automata, one for each token. Transitions within the sub-automata, in turn, are driven by the output of ART2 that determines the most likely set of templates for a given segment of the input. In operation, all fuzzy automatons operate and process inputs simultaneously. The operation of the HFA is discussed in a bottom up fashion, where at the bottom reside input templates, categorizations of the raw input, and at the top is the main automaton.

At the bottom of the hierarchy, the raw input is preprocessed, and then classified into the template classes. Template classes are assigned a membership based on parameters related to the goodness of classification, and several achieving good classifications are placed in the input template vector. For each input, a measure of fuzziness can be determined for all templates.

Sub-automata recognize tokens, sequences of templates, that are representative structures within the signal being analyzed. All transitions are initially assigned maximum fuzziness (i.e., 0.5), allowing any relevant template to produce a transition. State membership fuzziness is the maximum of two quantities, the first being the fuzzy membership of the destination state. The second quantity is the minimum of the membership of the

```
ART Categorization ART architecture adaptively forms categories using the signal segments pre-
sented. Details of the ART architecture may be found in [29].
ART Postprocessing

• select the number ($v = |V_k|$) of the categories (alphabet elements) representing each signal
  segment.

• do for each signal segment
      do for each category
          if (weight of current category > weight of any category in the set)
          then include the current category in the set
      replace the current signal segment by the current set of $v$ categories
```

Fig. 3. Pseudocode for ART postprocessing.

```
Syntactic Analysis

• do for each condition considered
      initialize $\omega_S(main)$ and $\omega_D(main)$
      do for each sub-automaton
          initialize $\omega_S(SA_i)$ and $\omega_D(SA_i)$
          initialize all states
              $\mu_{\phi_start} = 1.0$
              $\mu_{\phi_middle} = 0$
              $\mu_{\phi_end} = 0.5$
          initialize all transitions
      do for each set $\rho_k$
          do for each state $\phi_n$ in the main automaton
              do for each sub-automaton $SA_n$ that still can move to another state
                  do for each state $\phi_i$ in $\omega_S(SA_n)$
                      do for each alphabet element in $\rho_k$
                          if the alphabet element initiates a transition $\delta_{ij}$ from $\phi_i$
                              update $\omega_D(SA_n)$ by adding the destination state $\phi_j$ to $\omega_D(SA_n)$.
                              update $\mu_{\phi_j}$.
                              if $\phi_j$ is an end state
                                  add destination state of current transition in main automaton to $\omega_S(main)$.
                          else if no other transitions from $\phi_i$
                              drop $\phi_i$ from $\omega_S(SA_n)$
                  transfer all states in $\omega_D(SA_n)$ to $\omega_S(SA_n)$ for the next input.
                  $\omega_D(SA_n) \leftarrow \{\}$
                  if $\omega_S(SA_n)$ is empty (i.e., no more transition capability)
                      drop $\phi_n$ from $\omega_S(main)$
          update $\omega_S(main)$ by adding all states in $\omega_D(main)$
      find the minimum membership $\mu_{\phi_i,min}$ along the path of transitions
      declare if the condition is present and the accuracy computed.
```

Fig. 4. Pseudocode for the syntactic recognition.

initial state, and the template membership. This relationship ensures, within the current subautomaton, that certainty increases with unambiguous template memberships and that ambiguity is maintained with sustained ambiguity.

In Fig. 6, we illustrate a main automaton and three sub-automata used to analyze the signal shown in Fig. 5. Using the figures, we discuss the operation of HFAs. The analysis starts at all states in the main automaton. From Fig. 5, the main automaton can move from states one and three by the token A, from two by B, and from four by C. For each token that can move the main automaton from one state to another at the current time instant (i.e., for all currently active transitions), we use their representative sub-automata to determine whether the coming sequence of alphabet templates characterizes any of the

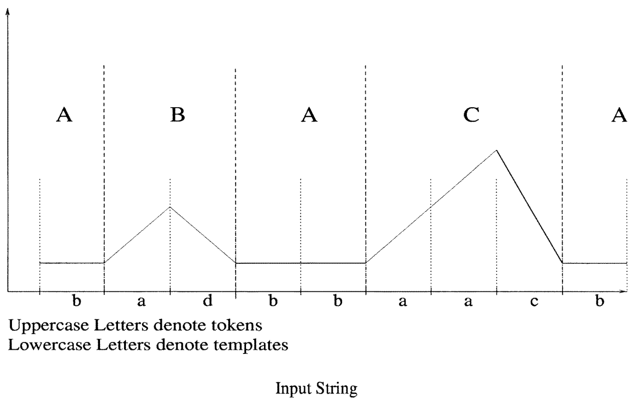
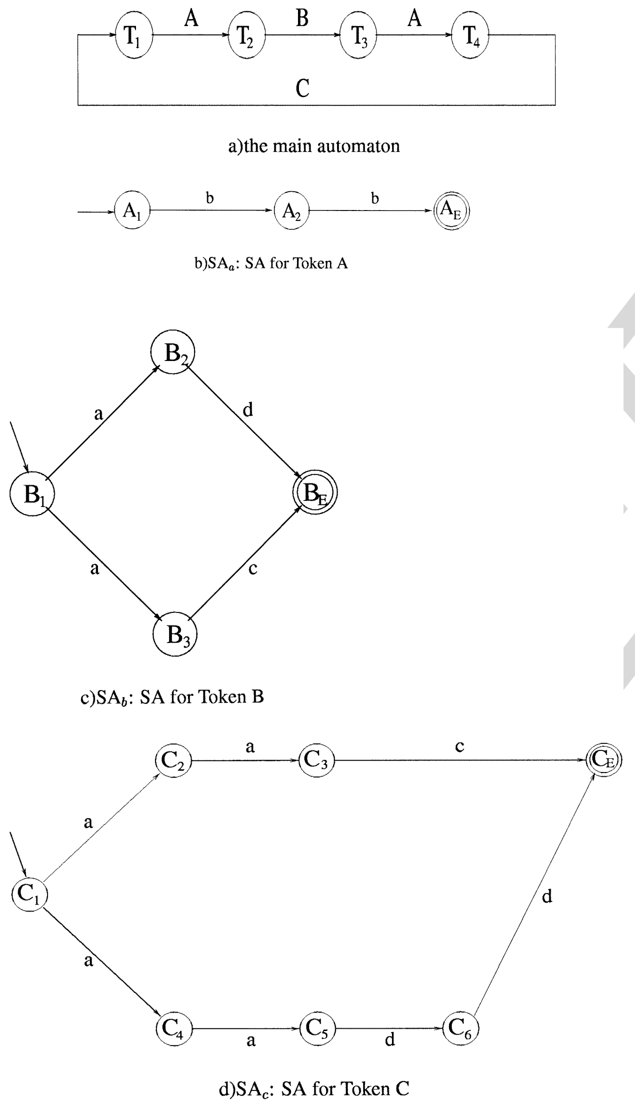


Fig. 5. Input signal shown as a sequence of alphabet templates.



$$\begin{tikzcd} T_1 \arrow[r, "A"] & T_2 \arrow[r, "B"] & T_3 \arrow[r, "A"] & T_4 \arrow[lll, bend left, "C"] \end{tikzcd}$$

$$\begin{tikzcd} {} \arrow[r] & A_1 \arrow[r, "b"] & A_2 \arrow[r, "b"] & A_E \end{tikzcd}$$

$$\begin{tikzcd} & B_2 \arrow[dr, "d"] & \\ B_1 \arrow[ur, "a"] \arrow[dr, "a"'] & & B_E \\ & B_3 \arrow[ur, "c"'] & \end{tikzcd}$$

$$\begin{tikzcd} & C_2 \arrow[r, "a"] & C_3 \arrow[rr, "c"] & & C_E \\ C_1 \arrow[ur, "a"] \arrow[dr, "a"'] & & & & \\ & C_4 \arrow[r, "a"'] & C_5 \arrow[r, "d"'] & C_6 \arrow[uur, "d"'] & \end{tikzcd}$$

Fig. 6. An example for input synchronization.

tokens. Initially, all states except the end state in sub-automata are initialized to a membership of 0.5. Thus, with the token, transitions may be taken from any but the end state. As a result, the state machine includes the correct synchronization state from the onset of processing, and does not require any predetermined point in the signal to start the analysis. Initialization and synchronization are discussed in more detail in Section IV.

Assume that input synchronization occurs at the second b in Fig. 5, and that the main automaton is in state T_3. Hence, subautomaton SA_a is the only active subautomaton. On the receipt of the template b, subautomaton SA_a moves from A_1 to A_2. Hence, the subautomaton SA_a remains active for the next template in the sequence. No changes occur in the main automaton since the token A, which is the input that initiates the transition from T_3 to T_4 in the main automaton, is not recognized yet. The next template presented is b again. The only active state in the main automaton is T_3, and the only active subautomaton represents A. The source state in the subautomaton is A_2. The template b can move the subautomaton SA_a from A_2 to A_E. A_E is the end state of the subautomaton SA_a. That is, the token A has been recognized by the subautomaton SA_a. This, in turn, moves the main automaton from T_3 to T_4. The new template presented is a. The only transition from T_4 in the main automaton can be initiated by the token C. The subautomaton SA_c becomes active. The subautomaton SA_c can move from the start state, C_1, to state C_2 via a. C_2 is not an end state. Hence, the subautomaton SA_c remains active and no changes occur in the main automaton. The next two templates are a and c, respectively, which move the subautomaton SA_c C_3 and to C_E, respectively. State C_E is the end state of the subautomaton SA_c. The moves from T_4 to T_1. This process proceeds until either all templates are presented to the system or a template in the sequence is rejected by the system.

III. Methodology for Constructing Diagnosis System

The diagnosis system described in the last section is constructed using some basic assumptions about the types of processing performed as well as, the characteristics of the signals in the class of systems for which automated diagnosis is desired. The diagnosis system is constructed in two stages, representing the system's independent and dependent parts, respectively. First, preprocessing and template identification are performed. Next, HFAs are constructed for each condition which is a largely manual process and is system dependent.

A. System Independent Methodologies

The systems under consideration are assumed to be measured and monitored through one or more time sampled analog signals. The signals are also assumed to have some temporal structure.

Preprocessing is the first step of the methodology. In preprocessing of the input signal, the following two tasks occur: the decomposition of the original signal into windows of time samples, the input segments, and the filtering and other adjustments applied to the input signal to improve the performance of the ART2 network.

The decomposition of the time-sampled signal into segments is necessary since ART2 can only accept the original signal as a sequence of input segments. A time-sampled signal is decomposed into segments by a sliding window where the window size determines the size of segments. The number of time samples the window leaves behind to obtain the next segment is another parameter of the signal segmentation process. Usually, different values of this parameter are used for training and testing (i.e., operation mode) purposes of ART2. To retrieve maximum information from the signal during the training period, a small number of time points may be selected to allow for overlapping. On the other hand, in operation mode, the sequence of segments must be identical to the original signal. Hence, no overlapping, as well as, no omission of any time point in the original signal is desirable during the operation of ART2.

During training, ART2 identifies alphabet templates. Alphabet identification serves as the boundary between the system independent and dependent methodologies. For different systems, the structural components may also vary as a function of the level of detail. At higher levels of signal detail, the components are weakly system dependent and general, while at lower levels of detail, the components are strongly system dependent and specific. In addition, the true nature of signals observed at higher levels of detail are more ambiguous, due to noise and other variations in the signal. The mechanism adopted in this work is to employ the ART2 ANN which can identify and then classify time sampled continuous measurements [29]. In this work, the Stuttgart Neural Network Simulator (SNNS) [31] and the accompanying ART algorithms have been modified to facilitate the creation of the fuzzy template vector used in subsequent processing.

In operation mode, ART2 transforms the original time-sampled signal into a sequence of vectors of alphabet templates. ART2 classifies each consecutive signal segment in the original signal with a vector of alphabet templates along with the corresponding strength of match. Over a sequence of segments, this new sequence is called a *multicategory string*. After classification, the trained ART2 network classifies the kth signal segment t_k in the original signal to each category i and assigns a weight w_{ki} reflecting the strength of the match between the segment t_k and the corresponding category i, represented by the alphabet template τ_i. Thus, the ART2 network generates a string s of templates out of an alphabet with p elements where each alphabet template τ_i has an associated weight w_{ki} for each signal segment t_k as in the following

$$s = \prod((\tau_1, w_{11}), \ldots, (\tau_p, w_{1p})), \ldots, ((\tau_1, w_{k1}), \ldots (\tau_p, w_{kp})), \ldots, ((\tau_1, w_{K1}), \ldots, (\tau_p, w_{Kp})) \quad (1)$$

where s is the multicategory string representing an input signal originally composed of K segments.

Examining all alphabet templates to detect the best match is time consuming, and degrades the system's computational performance. Furthermore, considering more than the few alphabet templates with the highest ranking weights does not improve the quality of the transformation. Hence, the multicategory string is optimized, such that a prespecified number v of alphabet templates, with the top-ranking weights is selected for each signal segment, and the string is formed by alphabet template vectors instead of the entire set of categories.

The multicategory string can be mathematically defined as follows: form a partition $\Sigma = V_k \bigcup T_k$ corresponding to each input signal segment t_k where $V_k \bigcap T_k = \{\,\}$ and $|V_k| = v$, and $|\Sigma| = p$. The resulting string is

$$s = S_1 \cdots S_K \quad (2)$$

where each vector S_k replaces a segment t_k and is a set of two-tuples, such that

$$S_k = \{(\tau_j, \mu_{kj}) | \, \forall j \, \forall i \, \tau_j \in V_k \wedge \tau_i \in T_k \wedge \mu_{kj} > \mu_{ki}\}. \quad (3)$$

The string s is composed of K vectors of templates, and μ_{kj} is the fuzzy membership assigned to the alphabet template τ_j to reflect the strength of the match with the signal segment t_k and is

$$\mu_{kj} = \mathbf{r}_{kj} \quad (4)$$

where $\mathbf{r}_{kj}$ is the match of kth signal segment and jth alphabet template derived from ART2.

The reason for a multicategory transformation (i.e., each signal segment t_k is replaced by a vector S_k of alphabet templates, and the corresponding top-ranking weights instead of only the one "winning" alphabet template τ_J with the highest weight) of the original signal is to convey more information to the syntactic recognizer, regarding alternative matches achieved by the operation of ART2, so that a possible miscategorization occurring in the decision-theoretic portion of the methodology can still be corrected in the syntactic recognizer. Creation of template alphabets is performed for each different condition for which diagnosis is desired, and the process of creating templates is relatively application independent.

B. System Dependent Methodologies

System dependent processing takes information about the characteristics of the system for creating fuzzy automatons. Identifying tokens from sequences of alphabet templates move down the next level of detail and up on the level of system specifics. Finally, strings of tokens are used to represent the different diagnosis conditions. Given a collection of automatons for the different diagnosis conditions, defuzzification into a crisp diagnosis is achieved.

HFAs perform a syntactic analysis on the string $s = S_1 \cdots S_K$, as in (2), and identify the condition of the target system. Each HFA that characterizes a condition under consideration (i.e., a condition which our methodology attempts to identify) is manually built. The manual construction of HFAs is discussed in Section IV-C.

In the next section, we discuss the two properties of the HFAs, the hierarchy and the input synchronization capability. Then, the construction of HFAs and the syntactic analysis are discussed.

IV. HFAs

An HFA is a finite state machine where states, transitions, and inputs are assigned fuzzy memberships. An input membership μ_{ki} reflects the strength of the match between the alphabet template τ_i, and the input segment t_k. A transition membership

$\mu_{\rho_{ij}}$ describes the fuzziness, or the uncertainty of moving from state ϕ_i to state ϕ_j. At any given time, the state machine can have a membership in several states simultaneously. The set of states from which an HFA can initiate transitions with the current input is the set ω_S of the current source states. The set of states to which the HFA progresses is called the set ω_D of current destination states. The HFA is at each of the states in ω_D only with a specific degree of certainty or fuzziness μ_{ϕ_i}.

In the process of recognition, an HFA moves from the fuzzy source states in ω_S, to the fuzzy destination states in ω_D, with each input S_k as in (3). At each transition, the membership of each destination state is updated based on the current membership of the source and destination state, the membership of the current input, and the transition membership. The memberships of the states, along the path from the start state to the end state, determine the strength of the recognition at the end of the fuzzy syntactic analysis. The transition memberships, on the other hand, remain the same throughout the analysis.

Each condition or fault for the system under analysis has an HFA associated with it. The HFA performs a syntactic analysis and determines with a specific accuracy whether the system under analysis indicates the condition associated with the HFA. An HFA for each condition consists of a main automaton and several sub-automata. The number of sub-automata for a condition is specified by the number of significant structures, the tokens, in the condition-specific signals which are used in the HFA construction and characterize the condition.

A. Hierarchy in HFAs

HFAs realize the syntactic analysis and recognition of a signal by a hierarchical mechanism. The main automaton recognizes the entire signal s, while each subautomaton achieves the recognition of a specific token in the signal. The inputs that can initiate transitions in the main automaton are the tokens that are recognized by sub-automata. Sub-automata, on the other hand, can progress from one state to another by alphabet templates. During the operation, the HFA receives the current vector of alphabet templates S_k in the multicategory string, and starts examining if any state in the current input storage $\omega_S(main)$ of the main automaton can initiate a transition with any of the alphabet templates in the current vector. However, the only way a transition can be initiated in the main automaton is by the recognition of a token in the string. Thus, the subautomaton corresponding to the token takes the syntactic analysis over to check whether the token that can make a transition from the current main automaton state can be recognized by the subautomaton. The subautomaton studies the source states in its ω_S to determine if a transition can be initiated from any of them, with any members of the current alphabet template vector S_k. The destination states at the end of the current transitions form ω_D of the subautomaton. If any of the destination states in ω_D are end states of the subautomaton, then the corresponding transition in the main automaton occurs, and the destination state of the transition in the main automaton is placed in ω_D of the main automaton. The above discussion explains the hierarchical interaction between the main automaton and the sub-automata for a specific condition. In the next section, we discuss the input synchronization property of HFAs.

B. Input Synchronization

In the methodology, no *a priori* assumptions are made about the starting position of an input signal. That is, the time sample at which the recording process of the original signal began may be at any point in any subautomaton. The methodology does not require that the first input be synchronized to any state machine in any way. The syntactic recognizer locates the correct position of the multicategory string during the analysis of the first few templates of the input string, and continues the analysis accordingly. The number of templates that the syntactic recognizer examines before the current position of the string is discovered depends upon the starting position of the string, and the shapes of tokens. If the string starts with a token that is composed of alphabet templates that are unique to this token, then the current position of the string is located immediately. If not, then the HFAs rule out the incorrect candidates for current positions when a token with a different shape is encountered. This capability of HFAs in the methodology is called *input synchronization*.

To achieve input synchronization, the syntactic analysis starts by having a nonzero membership in start and end states in all HFAs. Since all states are potential start states, all states form $\omega_S(main)$ of the main automaton in the beginning of the syntactic analysis. By considering all states in the main automaton, all tokens that can initiate transitions from states in $\omega_S(main)$ of the main automaton are examined. The examination of tokens proceeds with the analysis in sub-automata. Sub-automata are invoked by the main automaton as discussed in the previous section. The starting location of the string can be any point in the token. Therefore, all states but the end states in each subautomaton are put into the related initial ω_S. All states in ω_S of each subautomaton are checked to see if the presented alphabet template vector is able to initiate a transition. The destination states of the initiated transitions are stored in ω_D of each subautomaton. States with no possible transitions are dropped from ω_S. If a subautomaton cannot move from any of the states in the ω_S, then the state in the main automaton is dropped from ω_S of the main automaton from which the main automaton can only move by the token recognized by this specific subautomaton. For the analysis of the next alphabet template vector, the destination states in ω_D of each subautomaton are transferred to the next ω_S of the subautomaton and become the source state for the next alphabet template vector. At some point in the analysis, all sub-automata but one fail to proceed since, with the current alphabet template, none of the sub-automata can initiate a transition from any state. The remaining subautomaton reflects the token in the input string that is currently presented to the syntactic recognizer. If, before that point, all sub-automata terminate, then the input string is not recognized, or rejected by the HFA before even the current position of the string is located. We give a simple example to show how input synchronization works. We use the signal and the HFAs in Figs. 5 and 6, respectively. In Table I, we show how ω_Ss and ω_Ds of these HFAs change with each alphabet template in the sequence. For simplicity, we drop the fuzzy memberships of states, transitions,

TABLE I
INPUT SYNCHRONIZATION EXAMPLE

Alphabet Template	Main		SA A		SA B		SA C	
	ω_S	ω_D	ω_S	ω_D	ω_S	ω_D	ω_S	ω_D
b	$T1, \ldots, T4$	$T2, T4$	$A1, \ldots, A3$	$A2, A3$	$\{\}$	$\{\}$	$C1, \ldots, C6$	$C4$
a	$T1, \ldots, T4$	$\{\}$	$A2$	$\{\}$	$B1$	$B2, B3$	$C1, C4$	$C2, C$
d	$T2, T4$	$T3$	$\{\}$	$\{\}$	$B2, B3$	$B4$	$C2, C5$	$\{\}$
b	$T3$		$A1$		$\{\}$		$\{\}$	

and templates. The analysis starts at all states in the main automaton. Hence, the input storage of the main automaton initially is $\omega_S(main) = \{T1, \ldots, T4\}$. From Fig. 5, the first alphabet template is b. The two sub-automata A and C can start with b. Initially, $\omega_S(A) = \{A1, \ldots, A3\}$ and $\omega_S(C) = \{C1, \ldots, C6\}$. After the transitions, the output storages of the sub-automata, which are initially empty are updated as follows: $\omega_D(A) = \{A2, A3\}$, and $\omega_D(C) = \{C2, C3\}$. That is, A can make a transition from the first and second states and C only from the first state. The resulting states in A are the second state (transition from the first state), and the third state (transition from the second state). In C, the resulting state is $C3$. Since A reaches the end state ($A3$), ω_D of the main automaton is updated: $\omega_D(main) = \{T2, T4\}$.

For the next input, $\omega_S(main) = \{T1, \ldots, T4\}$. The first and third state of the main automaton are still active, and the second and fourth states are copied from $\omega_D(main)$ of the last transition. Thus, $\omega_S(A) = \{A2\}$, and $\omega_S(C) = \{C3\}$. The new added second state can initiate a transition with B: $\omega_S(B) = \{B1\}$. The fourth state can initiate a transition with C. Thus, $\omega_S(C) = \{C1\}$. The second input is a from Fig. 5. The template a cannot initiate a transition in A from any states in the current $\omega_S(A)$. Hence, $T1$ and $T3$ drop from $\omega_S(main)$. B can initiate transitions to $B2$ and $B3$. Hence, $\omega_D(B) = \{B2, B3\}$. C can initiate transitions from $C1$ and $C3$ to $C2$ and $C5$, respectively. Hence, $\omega_D(C) = \{C3, C5\}$. Since no sub-automata reached their end states, $\omega_D(main) = \{\}$.

For the next input, $\omega_S(main) = \{T2, T4\}$. $T2$ and $T4$ are still active in the main automaton. No state is copied from $\omega_D(main)$ of the last transition. Hence, $\omega_S(B) = \{B2, B3\}$ and $\omega_S(C) = \{C3, C5\}$. The next input is d from Fig. 5. The template d cannot initiate a transition in C from $C3$ or $C5$ in the current $\omega_S(C)$. Hence, $T3$ drops from $\omega_S(main)$. B can initiate one transition to $B4$ from $B2$. Hence, $\omega_D(B) = \{B4\}$. Since B reached its end state, the resulting state of B in the main automaton ($T3$) is put into the output storage. Thus, $\omega_D(main) = \{T3\}$.

For the next input, $\omega_S(main) = \{T3\}$. The first state of the main automaton is still active, and no state is copied from $\omega_D(main)$ of the last transition. Thus, $\omega_S(A) = \{A1\}$. The next input is b from the Fig. 5. On the receipt of the fourth input, only one state remains in $\omega_S(main)$, and the operation has synchronized with the input string (i.e., the next token in Fig. 5 is A, and it is the one before C).

After this discussion of the properties of the HFAs, we present a discussion regarding the construction of the HFAs.

C. HFA Construction

In this section, we discuss how an HFA for a condition is developed. A condition is the status of the system under analysis, including fault free and faulty operation. Furthermore, to analyze a system for a condition, an HFA must be constructed corresponding to the condition. From the viewpoint of the automated electrocardiogram (ECG) diagnosis, the condition is the healthy, or pathologic status of the heart. To construct an HFA for a condition, a signal set characterizing the condition is transformed by the ART2 ANN into a sequence of template vectors, one signal at a time. Next, the transformed signal set is used to build the HFA. A set of signals, the structures of which clearly reflect typical features of the condition, allows the construction of an HFA that provides for robust recognition. To reach a reliable diagnosis in multiple-condition situations, the individual accuracies the methodology provides for each of these conditions may be examined at the end of the analysis. Another way to handle the complex problem of a multiple-condition situation may be to check signals produced by other possible sources within the same system which may enhance the features of one condition while suppressing those of the other conditions. Another possible solution may be to use a more sophisticated defuzzification technique that can relate different conditions to each other and extract a more accurate diagnosis. The last solution is beyond the scope of this study.

The HFA construction for a condition starts when a multicategory string of alphabet templates is obtained for each sample signal in the set. First, the tokens representing significant structures in the sample string are identified. Next, a transition diagram of the main automaton is constructed where each transition of the main automaton is assigned the corresponding token. Each state in the main automaton is assigned a fuzzy membership of 1 since all states in the main automaton are the start states enabling input synchronization. The fuzzy membership of a transition in the main automaton is the fuzzy membership of the end state of the corresponding subautomaton that recognizes the token initiating the transition. The initial values for the transitions in the main automaton are 1. As noted earlier, the main automaton is built cyclicly, thus capturing the periodicity of the signals under analysis, and with no explicit end, states to accept signals of any length.

To construct the subautomaton for a given token, all occurrences of the corresponding token are identified. Each occurrence is actually a sequence of alphabet templates. Next, a path of states is inserted to the subautomaton which is used to recognize this specific sequence of alphabet templates. This insertion is performed so as to provide the minimum complexity and maximum flexibility of the state machine. The fuzzy memberships of the transitions can either be obtained from *a priori* knowledge of the transitions or initialized to the maximum uncertainty, 0.5, in the case of no prior information.

In the case of no prior information, an initialization with memberships reflecting the maximum ambiguity is used to allow any possible transition that occurs in the subautomaton. Moreover, this initialization could be a good initial point to adaptively determine the transition memberships. The state memberships change during the process of recognition. Synchronization occurs when all but one of these states fails to continue, after a few transitions, since these states are not the correct states at which the subautomaton starts with the given input string. To make the input synchronization mechanism work, the start state of a subautomaton is assigned 1, the end states 0.5, and the other states (the pseudo-start states) 0.

We show the HFA construction process by means of a simple example illustrated in Figs. 5 and 6. Fig. 6 is used in this example. The construction of the HFA starts with the identification of tokens in the sample signal by a human expert. Suppose one period of our sample signal is illustrated in the input string in Fig. 5. We can observe from the figure that one period of the signal is composed of the tokens A (the horizontal neutral line), B (the small triangular wave), and C (the larger, irregular triangular wave). The sequence is A, B, A, C. Next, we draw the transition diagram for the main automaton using the sequence of tokens in one period. Then, the human expert identifies the sequence of winning alphabet templates (a winning alphabet template is one that matches the corresponding window of samples with the highest weight) in the transformed signal.

The postprocessor generates the string of template vectors and the categorization that the ART2 ANN performed for each segment window, and the winning alphabet template that the postprocessor finds by means of the mean square error (MSE) criterion. The template alphabet in Fig. 6 is $\Sigma = \{a, b, c, d\}$. Suppose our sample signal consists of 1000 time samples, the window size is $w = 10$, and the sliding step $s = 10$. With this configuration we have 100 windows. Suppose that the waveform shown in Fig. 5 is the start of the sample signal. Then, the first part of the string appears, as in Table II. Note that categories that ART2 suggested are omitted in Table II for convenience. Using the input string in Fig. 5 and Table II, we can determine which alphabet templates each token is composed of, and start to construct the sub-automata for tokens.

As an example, we consider token B. In the input string in Fig. 5, B is contained in the second and third windows. Table II shows that the winning templates for windows 2 and 3, are a and d, respectively. This means that a path should exist in the subautomaton for B that transitions the subautomaton from the start state to the end state by the consecutive transitions initiated by a and d. Then, we draw the upper path in the transition diagram of the subautomaton for B. We continue identifying other Bs in the sample signal, and add other possible paths to the subautomaton for B in the same fashion. Suppose that we identify B in windows 33 and 34, 46 and 47, and 69 and 70 in the sample signal, and the string is as in Table III. Suppose that the only sequence of winning templates that is different from the first sequence of templates that identify the token B (see Table III) is that for windows 46 and 47, where the second winning template is c. Hence,

TABLE II
FIRST PORTION OF THE TEMPLATE VECTOR STRING FOR THE INPUT SIGNAL

Window	1	2	3	4	5	6	7	8	9
Winning Template	b	a	d	b	b	a	a	c	b

we add this path to the subautomaton of B (the lower path in the subautomaton for token B) in Fig. 6. When all different versions of B in all sample signals are added to the transition diagram of the subautomaton, we combine the states which are equally far from the start states, and to which the subautomaton arrives by transitions initiated by the same templates. In the case of B, we can combine the states 2 and 3, since they are both one transition from the start state, and reached by transitions initiated by a, as shown in Fig. 7. To cope with imperfect signals and to tolerate the possible incorrect classifications by the ART2 ANN, additional transitions can be considered simultaneously between any two states which are performed by templates similar to the winning template. The similarity criterion is MSE with respect to the winning template. The number of additional transitions considered depends upon the application and the MSE distance between the winning and similar templates.

Following the procedure used for B, we perform the same process for all other tokens in all sample signals and construct the corresponding sub-automata. Finally, we assign the fuzzy memberships to the states and the transitions as discussed above. This completes the example and the discussion of the HFA construction.

D. Diagnosis

The syntactic recognizer, performs the recognition in the diagnosis phase. The syntactic recognizer consists of several HFAs. For each condition ψ_j in the system, there exists one HFA that recognizes the strings $s = \prod_{k=1}^{K} S_k$. At the end of the training of ART2, each condition has an alphabet $\Sigma(\psi_j)$. For each alphabet, there exists an HFA. The main automaton performs transitions upon receipt of a token. For the transition to be performed in the main automaton by a token, a subautomaton for the corresponding token must recognize the token from the coming templates. This emphasizes the hierarchical nature of the syntactic recognition process. To perform a transition, HFAs receive the current set of alphabet templates, S_k as in (3), at each step of execution from the input signal $s = S_1 \cdots S_K$. HFAs consider all of these alphabet templates to concurrently perform transitions. For each HFA, an ω_S and an ω_D hold the source and resulting states for the current transitions, respectively. For a transition to occur from a source state, the membership of the current input (a token in the main automaton, and an alphabet template in subautomaton) must be greater than, or equal to, the membership of the transition. If this condition is fulfilled, the membership of the resulting state assumes the minimum value of the membership of the source state of the transition, and the membership of the current input. Otherwise, the membership of the resulting state stays the same. If there exists more than one possible transition from states in ω_S to the resulting state with alphabet templates in the same set S_k, then the membership of the resulting state is

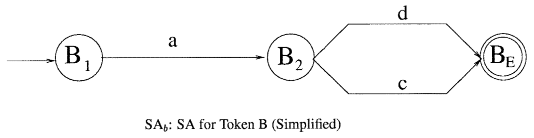


Fig. 7. Simplified diagram of the SA recognizing token B.

assigned the maximum among those. In other words, the state membership update function is expressed as

$$\mu_{\phi_j}^{k+1} = \begin{cases} \max\limits_{i \in I_s^k} \left(\mu_{\phi_j}^k, \min\left(\mu_{S_{\delta_{ij}}^k}, \mu_{\phi_i}^k \right) \right), & \text{if } \mu_{\delta_{ij}} \le \mu_{S_{\delta_{ij}}^k} \\ \mu_{\phi_j}^k, & \text{otherwise} \end{cases} \tag{5}$$

where the superscript k denotes discrete time steps, ϕ_i is a source state, ϕ_j is a destination state, δ_{ij} is the template that must be input to transition from state ϕ_i to ϕ_j, $S_{\delta_{ij}}^k$ a template derived by ART2 classifications, μ_{ϕ_i} is the fuzzy state membership in state ϕ_i, $\mu_{\delta_{ij}}$ is the fuzzy transition membership from ϕ_i to ϕ_j, $\mu_{S_{\delta_{ij}}^k}$ is the membership of the template, and

$$I_s^k = \left\{ \phi_i | \phi_i \in I_j \wedge S_{\delta_{ij}}^k \in \mathbf{S}^k \right\} \tag{6}$$

where I_j is the set of states that can transition to state ϕ_j.

We give an example in Fig. 8 to clarify the function of the formula. There are two states 1 and 2 from which the automaton can move to state 3. Both transition memberships are $\mu_{\delta_{1,3}} = \mu_{\delta_{2,3}} = 0.5$. One instant (discrete time k) from the automaton is illustrated in Fig. 8, at which the state memberships of state 1 and 2 are $\mu_{\phi_1} = 0.7$ and $\mu_{\phi_2} = 0.8$, in respective order, and the corresponding input memberships are $\mu_{S_{\delta_{1,3}}^k} = 0.6$ and $\mu_{S_{\delta_{2,3}}^k} = 0.9$. We first check if the condition on the righthand side of (5), which requires that input memberships must be greater than the transition memberships. For both transitions, the current input memberships are greater than 0.5, the transition memberships. Hence, the condition on the righthand side is fulfilled. Next, we take the minimum of the initial state and input memberships. For transition $1 \rightarrow 3$, $\min(0.6, 0.7) = 0.6$, and for transition $2 \rightarrow 3$, $\min(0.9, 0.8) = 0.8$. Then, for each transition we compute the maximum among the current destination state membership, and the result of the min operation. For transition $1 \rightarrow 3$, $\max(0.6, 0.6) = 0.6$, and for transition $2 \rightarrow 3$, $\max(0.6, 0.8) = 0.8$. Finally, we select the maximum among the results of the maximum operations for each transition to the current destination state, $\max(0.6, 0.8) = 0.8$. Hence, $\mu_{\phi_j}^{k+1} = 0.8$.

This example completes this section. We discuss the defuzzification process in Section IV-E.

E. Defuzzification: Determining the Likely Condition

In the last stage of processing, defuzzification is performed. While a fuzzy membership can be extracted from the HFA, it was found that a better measure of overall HFA success was a tally of the number of consecutive transitions within the state machine structure that are made over all input. This serves as a synchronization measure to measure of how much of the input followed the HFA. Using this as a basis, if several machines are able to synchronize on the input, more transitions indicate a better match. In the event multiple HFAs give the same, or very closely the same number of transitions, then the machine fuzziness is used to break the tie. In a sense, the diagnosis is the condition associated with the HFA that best synchronizes with the input recording.

V. Example: ECG Diagnosis

In this example, a demonstration of automatic diagnosis of the ECG is presented. Two HFAs are constructed to process, respectively, the normal sinus rhythm, and the atrial fibrillation, a type of arrhythmia originating from the atria. A normal sinus rhythm is illustrated in Fig. 9. In the normal sinus rhythm, the simplified cardiac cycle is composed of the P wave, QRS complex, and the T wave followed by the constant isoelectric line separating two cardiac cycles. Fig. 10 shows an atrial fibrillation which is an abnormal heart rhythm. In atrial fibrillation, the ECG exhibits an undulation of the baseline, called the fibrillatory, or the f-waves, accompanied by an irregular ventricular rhythm. While the QRS complex and the T wave keep the normal configuration, an atrial fibrillation rhythm might look like $ffQRSTffffQRSTfQRSTfffQRSTffffff f$. The features of atrial fibrillation can be best detected in the lead V_1, or the standard lead II [32].

A. Template Alphabet Construction

In this example, two ART2 networks and template alphabets are created for the normal sinus rhythm and atrial fibrillation. The task of the preprocessor is the decomposition of the input ECG. The preprocessor accomplishes the decomposition with a moving window.

The two parameters of the window, the window size w, and the slide step s, directly affect the performance of the automatic diagnosis. Care should be exercised while selecting the two parameters so that as much information as possible is transferred to the syntactic recognizer while avoiding a selection that would slow the analysis. To transfer maximum possible information, overlapping of the consecutive input segments is allowed in the decomposition of the time-sampled signal. For an overlap to occur, the window size must be strictly greater than the slide step. A slide step of one time sample gives the largest overlap and, hence, the transfer of maximum information. Furthermore, the slide step of one sample provides more gradual transitions between tokens. On the other hand, large overlaps slow the analysis. If the slide step and the window size are the same, the sequence of consecutive windows of time samples becomes an exact copy of the original signal, only decomposed into segments. If the slide step is greater than the window size, a loss of information occurs since some time samples are omitted.

The window size is another factor that affects the speed of the analysis. Smaller window sizes result in smaller template sizes, speeding up the tokenization process by the ART2 network. Choosing a smaller window size with larger overlaps appears reasonable, since the small window size may have a compensating effect on a large overlap by slowing the transformation of the signal. However, small window sizes cause the trans-

TABLE III
COMPLETE TEMPLATE VECTOR STRING FOR THE INPUT SIGNAL

Window	1	2	3	4	5	6	7	8	9	⋯	33	34	⋯	46	47	⋯	69	70
Winning Template	*b*	*a*	*d*	*b*	*b*	*a*	*a*	*c*	*b*	⋯	*a*	*d*	⋯	*a*	*e*	⋯	*a*	*d*

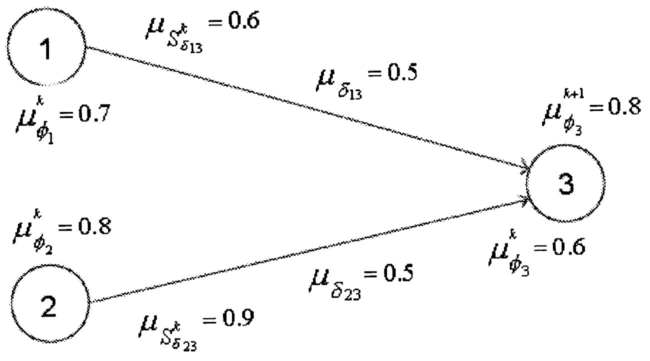

1
2
3
$\mu_{S^k_{\delta_{13}}} = 0.6$
$\mu_{\delta_{13}} = 0.5$
$\mu^k_{\phi_1} = 0.7$
$\mu^{k+1}_{\phi_3} = 0.8$
$\mu^k_{\phi_2} = 0.8$
$\mu^k_{\phi_3} = 0.6$
$\mu_{\delta_{23}} = 0.5$
$\mu_{S^k_{\delta_{23}}} = 0.9$

Fig. 8. Example showing the execution of the formula.

formed signals and the tokens to become longer. As a result, HFAs that recognize the tokens and the transformed signals become more complicated, and the syntactic analysis of longer signals by more complicated HFAs degrades the overall computational performance of the diagnostic system.

After a series of conducted experiments, the two parameters of the window, the window size w and the slide step s, have been selected to be $w = 10$ and $s = 2$ during training. We have used $s = 2$ since we have not observed any improvement in the system performance with the case $s = 1$ that provides the maximum information transfer to the syntactic recognizer. Further, the window size $w = 10$ has provided less HFA complexity for the same system performance as those obtained with larger window sizes (i.e., more complex HFAs).

For the HFA from atrial fibrillation, an ART2 network is trained with 600 templates, 200 of which originate from three different patients. For the HFA from normal sinus rhythm, an ART2 network is trained with 200 templates originating from one patient. Each template consists of 10 consecutive time samples. For the normal sinus rhythm, an ART2 network with 83 categories (alphabet templates) is trained. The ART2 network trained for atrial fibrillation contains 87 categories. Plots showing this template alphabet can be found in [27] and [30].

B. HFA Construction Phase

The first task is to determine the structure of the main automaton and each subautomaton to recognize an ECG signal. The main automaton is used to recognize a full cardiac cycle. In the ECG signal analysis, each token represents either a complex or a wave. Each subautomaton recognizes a part of the signal structure specific to that condition. For instance, in a normal sinus rhythm, a subautomaton is designed to recognize each of the tokens (P wave, the PR interval, the QRS complex, the ST segment, the T wave, and the neutral line showing the quiescent period between two consecutive heart beats), while for the atrial fibrillation, subautomaton are designed to recognize the tokens (the QRS complex, the ST segment, the T wave, and the TQ interval) since atrial fibrillation changes the normal sinus rhythm, and generates complexes with characteristics of atrial fibrillation. The transition diagram of the main automaton for the normal sinus rhythm is illustrated in Fig. 11. The sub-automata for the QRS wave and the ST segment of the ECG are shown in Figs. 12 and 13.

VI. RESULTS

Forty ECGs indicating atrial fibrillation and 18 ECGs indicating normal sinus rhythm are used in the experiments. Three ECGs have been used to train the ART2 networks and build the HFAs for the atrial fibrillation case. The remaining 37 are used to test the diagnosis methodology. One out of 18 ECGs were used to train the ART2 networks and construct the HFAs for the normal sinus rhythm. The remaining 17 ECGs have been used to test the method. Furthermore, ten atrial flutter cases have also been presented to both HFAs. The standard medical diagnostic parameters are used to evaluate the results [7]. Definitions of these parameters and our results are summarized in Tables IV and V.

The pair of HFAs has been able to distinguish correctly 35 out of 37 ECGs recorded from patients suffering from atrial fibrillation. Hence, the sensitivity of the HFA approach is 0.95. Furthermore, nine out of seventeen normal sinus rhythms have been correctly distinguished by the HFAs. Eight of the ECGs with normal sinus rhythm have been incorrectly diagnosed to have atrial fibrillation. Both HFAs (of the normal sinus rhythm and the atrial fibrillation) recognized five out of these eight ECGs with very close accuracies. For these five ECGs, the final minimum membership value of the atrial fibrillation was slightly higher than that of the normal sinus rhythm. Finally, both HFAs built to recognize the atrial fibrillation and normal sinus rhythm have recognized none of the atrial flutter cases.

VII. DISCUSSION AND CONCLUSION

The automatic diagnosis methodology we have presented here has been evaluated based upon the number of correct diagnoses among all ECG signals presented to the system. We have used the standard medical diagnostic accuracy criteria defined in Table V used by systems attempting medical diagnosis. Unlike our automatic diagnostic system, many signal processing algorithms attempting automatic detection, or recognition of a specific heart condition assess the performance of their methods by the number of the heart cycles recognized over the total number of cycles presented to the system. We chose to evaluate our system's performance based on the standard medical diagnostic accuracy criteria, since our system's final decision will be the identification of a specific heart problem using an ECG, rather than only a cycle identification.

Given the fact that the test data used in the experiments conducted were totally unknown to the diagnostic system, the diagnostic system displayed a considerably high reliability on distinguishing atrial fibrillation from the normal sinus rhythm (Sen-

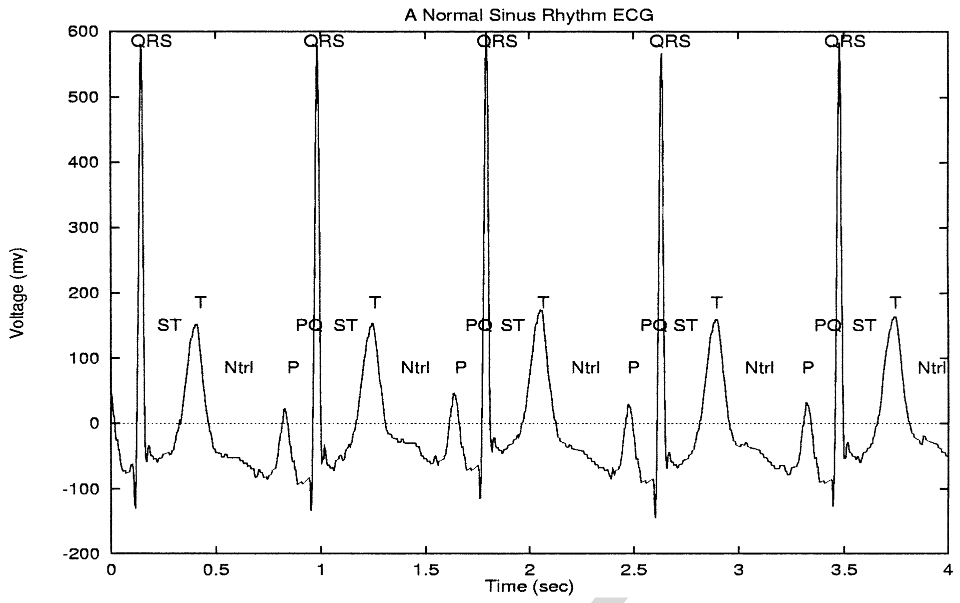


Fig. 9. ECG illustrating normal sinus rhythm.

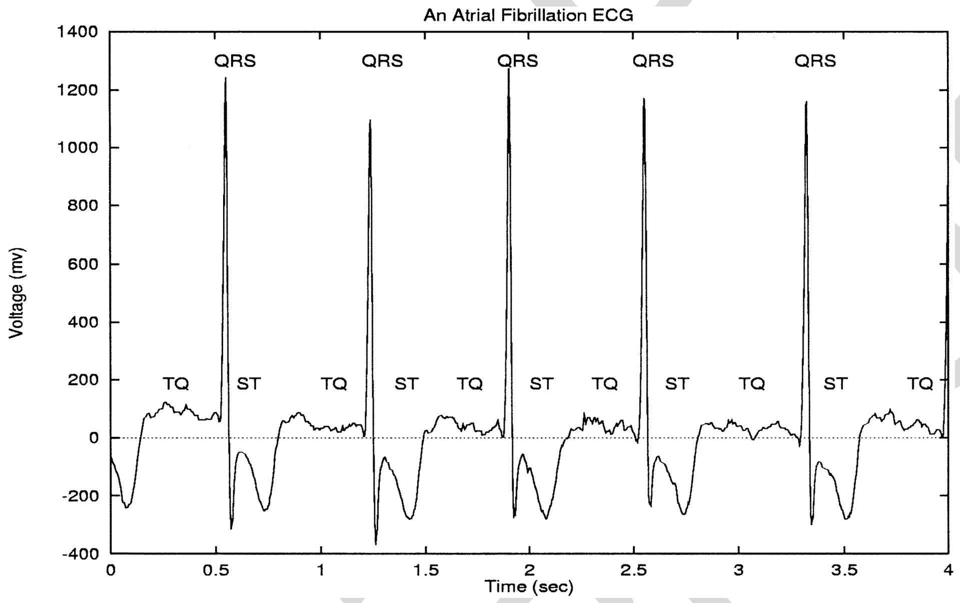


Fig. 10. ECG illustrating atrial fibrillation.

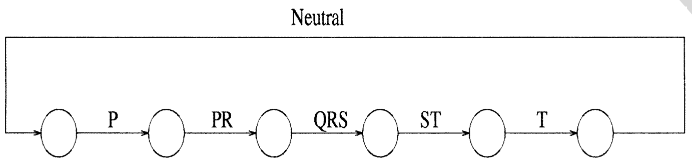


Fig. 11. Main automaton recognizing an ECG with normal sinus rhythm.

sitivity: 0.95). The specificity was not as high as the sensitivity. The primary reason for this is the diverse structure of the QRS complex in various ECGs. Using real time ECG signals, we encountered QRS complexes with various shapes. This was due to the fact that the corresponding lead was placed at a variety of places on the human chest and the direction of the electrical force was recorded accordingly. In such a case, the neural network solution to the problem of obtaining the signal primitives for a specific heart condition (i.e., the utilization of ART2 architecture to transform the original input signal into a sequence of vectors of primitive categories) requires that primitives composing many variants to the same token (in our case, QRS complex) in various signals of the heart condition be all included in the training set of the neural network. Further, these primitives should all be used in the manual construction of the related subautomaton. Due to the lack of sufficient data, we were able to establish only an incomprehensive subautomaton to characterize the QRS complex.

The time complexity of the algorithm depends on the following five factors:

1) the number α of the heart conditions under analysis;
2) the number τ of signal segments in the original input signal;

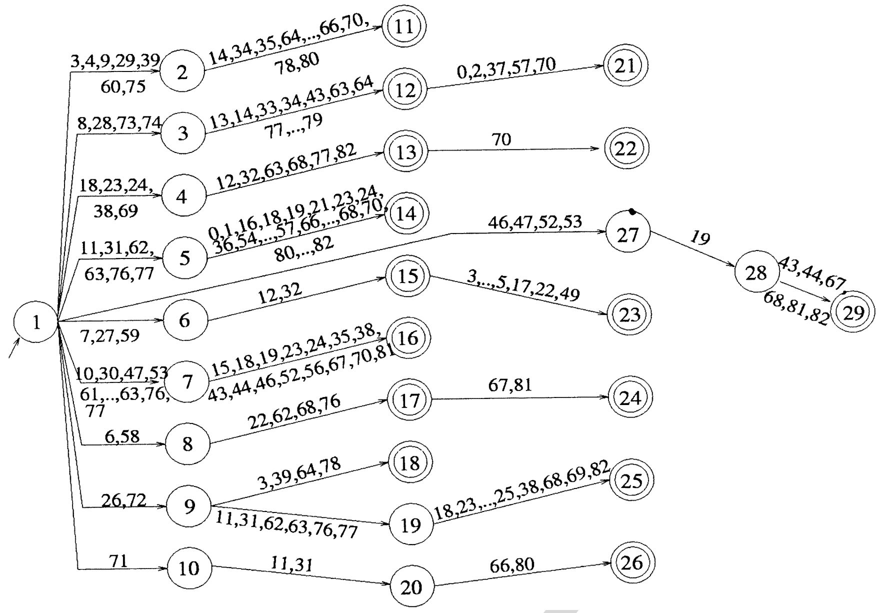


Fig. 12. SA recognizing a QRS complex in the normal sinus rhythm.

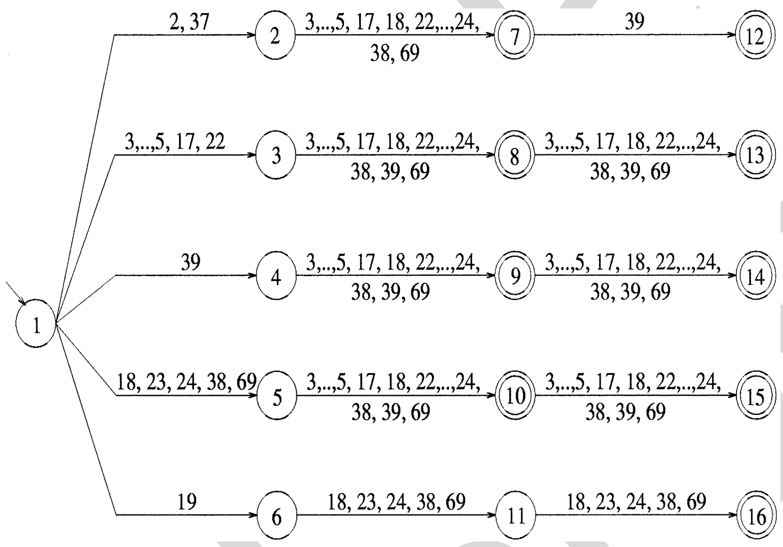


Fig. 13. SA recognizing the ST segment in the normal sinus rhythm.

TABLE IV
TABULATIONS OF TEST RESULTS

	Disease Present	Disease Absent	Total
Test positive	a=35	b=8	$a+b$=43
Test negative	c=2	d=9	$c+d$=11
Total	$a+c$=37	$b+d$=17	$a+b+c+d$=54

TABLE V
STANDARD ACCURACY CRITERIA AND RESULTS

Measure Name	Definition	Actual Value
Sensitivity	$\frac{a}{a+c}$	0.95
Specificity	$\frac{d}{b+d}$	0.53
False Negative Rate	$\frac{c}{a+c}$	0.05
False Positive Rate	$\frac{b}{b+d}$	0.47
Predictive Value positive	$\frac{a}{a+b}$	0.81
Predictive Value negative	$\frac{d}{c+d}$	0.90
False Alarm Rate	$\frac{b}{a+b}$	0.19
False Reassurance	$\frac{c}{c+d}$	0.18

3) the number v of alphabet elements that replace a segment in the original signal;
4) the number S of states in the current input set of states (the worst condition: all states);
5) the number T of transitions initiated from the current state (the worst condition: all transitions).

The decision-theoretic component of the diagnostic system is the system-independent part and has two operating modes: 1) the training of the ART2 neural architecture [29], and forming the alphabet of signal primitives, and 2) the transformation of original input signals into a multicategory sequence alphabet primitives, in which each signal segment in the original input signal segment is replaced by a vector of signal primitives in the alphabet. The second component is the HFAs that achieve the syntactic recognition.

The training of the ART2 architecture takes considerably less time, compared with other classical neural network architectures such as, backpropagation due to its bidirectional interaction between the input and output layers [31]. It took 683 s to train an ART2 neural architecture with signal segments of 10 time points (i.e., the input layer has 10 elements) and 102 categories. In another experiment, it took 1032 s to train an ART2 neural architecture with signal segments of 10 time points and 118 categories. We conducted both experiments on an Ultra-5 Sun Workstation running a Unix SunOS 5.7 operating system.

During the syntactic recognition, HFAs perform a check for each signal segment τ in each vector υ in the multicategory representation of the original signal. The number of checks, thus, equals to the product $\tau * \upsilon$. Each check is composed of determining all transitions T in each currently active state S of the HFA. The worst case scenario is that the active set of states is all states in the HFA representing a token. The basic operation consists of determining the active states of each HFA, and moving the HFA to these states (i.e., forming the new current input set of states of the HFA using these states). Finally, this process is performed for each heart condition under analysis. Hence, the execution time of the syntactic analysis can be found as follows:

$$t = \alpha \times \tau \times \upsilon \times S \times T. \tag{7}$$

Using the same machine specified above, it took 4.375 s to process an input file with 15 000 time points for two conditions. Another ECG signal of 12 000 time points was analyzed in 3.569 s by the syntactic recognizer. From the time complexity viewpoint, these figures show that this methodology is even suitable for use in emergency cases once the system-independent component (i.e., the ART2 neural net and the alphabet of the signal primitives) is constructed for each desired heart condition.

VIII. Summary

This paper presents the use of HFAs as a diagnostic tool for nonlinear systems. In this paper, HFAs were defined in terms of structure and function. HFAs use a fuzzy syntactic approach for diagnosis of time-sampled signals. In operation, the HFA transforms the signal into a string of sets of elementary structures, templates. Then, examining the consecutive templates, the HFAs determine whether or not the string characterizes a particular condition. After the syntactic analysis is performed for each desired condition, state synchronizing measures, and state memberships from these HFAs are used to identify the condition. The HFAs were applied to the problem of ECG diagnosis with good results.

References

[1] M. Bodo, G. Thuroczy, I. Nagy, J. Peredi, K. Sipos, P. Harcos, Z. Nagy, J. Voros, L. Zoltay, and L. Ozsvald, “A complex cerebrovascular screening system (cerberus),” *Medical Progress Through Technology*, vol. 21, pp. 53–66, 1995.

[2] J. Nieberl, S. Khoor, M. Khoor, E. Kekes, and F. Szaboki, “Twin expert systems for the complex analysis of conventional 12 lead electrocardiograms,” *Computers Cardiology*, pp. 511–513, 1993.

[3] A. Taddei, M. Niccolai, M. Emdin, and C. Marchesi, “A knowledge-based system for the interpretation of arrhythmias in long term ECG,” *Computers Cardiology*, pp. 887–890, 1993.

[4] Q. Xue, Y. H. Hu, and W. J. Tompkins, “Neural-network-based adaptive matched filtering for qrs detection,” *IEEE Trans. Biomed. Eng.*, vol. 39, pp. 317–329, April 1992.

[5] **AU: WHICH JOURNAL IS NEU?**J. J. Choi, K. H. O’Keefe, and P. K. Baruah, “Nonlinear system diagnosis using neural networks and fuzzy logic,” *Neu.*, vol. 20, pp. 813–819, 1992.

[6] Y. H. Hu, W. J. Tompkins, and Q. Xue, “Artificial neural network for ECG arrhythmia monitoring,” *Neu.*, vol. 20, pp. 987–991, 1992.

[7] B. W. Jervis, M. R. Saatchi, A. Lacey, T. Roberts, E. M. Allen, N. R. Hudson, S. Oke, and M. Grimsley, “Artificial neural network and spectrum analysis methods for detecting brain diseases from the CNV response in the electroencephalogram,” *IEE Proc. Science, Meas. Technol.*, vol. 141, pp. 432–440, 1994.

[8] E. Pietka, “Feature extraction in computerized approach to the ECG analysis,” *Pattern Recognit.*, vol. 24, pp. 139–146, 1991.

[9] Y. Suzuki and K. Ono, “Personal computer system for ECG ST-segment recognition based on neural networks,” *Med. Biol. Eng. Comput.*, vol. 30, pp. 2–8, 1992.

[10] T. F. Yang, B. Devine, and P. W. Macfarlane, “Artificial neural networks for the diagnosis of atrial fibrillation,” *Med. Biol. Eng. Comput.*, vol. 32, pp. 615–619, 1994.

[11] S. Barro, R. Ruiz, D. Cabello, and J. Mira, “Algorithmic sequential decision making in the frequency domain for life threatening ventricular arrhythmias and imitative artifacts: A diagnostic system,” *J. Biomed. Eng.* , vol. 11, pp. 320–328, 1989.

[12] M. Juhola and T. Gronfors, “A scheme of inference of regular grammars for the syntactic pattern recognition of saccadic eye movements,” *Artif. Intell. Med.*, vol. 3, pp. 87–93, 1991.

[13] A. Koski, M. Juhola, and M. Meriste, “Syntactic recognition of ECG signals by attributed finite automata,” *Pattern Recognit.*, vol. 28, pp. 1927–1940, 1995.

[14] P. Trahanias, E. Skordalakis, and G. Papakonstantinou, “A syntactic method for the classification of the QRS patterns,” *Pattern Recognit. Lett.*, vol. 9, pp. 13–18, 1989.

[15] P. Trahanias and E. Skordalakis, “Syntactic pattern recognition of the ECG,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 648–657, July 1990.

[16] G. Belforte, R. DeMori, and F. Ferraris, “A contribution to the automatic processing of electrocardiograms using syntactic methods,” *IEEE Trans. Biomed. Eng.*, vol. 26, pp. 125–136, 1979.

[17] S. L. Horowitz, “A syntactic algorithm for peak detection in waveforms with applications to cardiography,” *Commun. ACM*, vol. 18, pp. 281–285, 1975.

[18] E. Skordalakis, “Syntactic ECG processing: A review,” *Pattern Recognit.*, vol. 19, pp. 305–313, 1986.

[19] J. K. Udupa and I. S. N. Murthy, “Syntactic approach to ECG rhythm analysis,” *IEEE Trans. Biomed. Eng.*, vol. 27, pp. 370–375, 1980.

[20] F. Steimann and K. P. Adlassnig, “Clinical monitoring with fuzzy automata,” *Fuzzy Sets Syst.*, vol. 61, pp. 37–42, 1994.

[21] A. Bugarin, S. Barro, R. Ruiz, J. Presedo, and R. Palacios, “Syntactic characterization of qrs morphology,” in *Annual Intl. Conf. IEEE Eng. Med. Biol. Soc.*, 1991, pp. 588–589.

[22] T. Kalayci and O. Ozdamar, “Wavelet preprocessing for automated neural network detection of eeg spikes,” *IEEE Eng. Med. Biol. Mag.*, pp. 160–166, Mar./Apr. 1995.

[23] A. V. Oppenheim, A. S. Willsky, and I. T. Young, *Signals and Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[24] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[25] M. M. Gupta, G. N. Saridis, and B. R. Gaines, *Fuzzy Automata and Decision Processes*. Amsterdam, The Netherlands: Elsevier, 1977.

[26] A. Kandel and S. C. Lee, *Fuzzy Switching and Automata*. Ann Arbor, MI: Edwards, 1979.

[27] M. B. Tümer, L. A. Belfore, II, and K. M. Ropella, “Applying hierarchical fuzzy automatons to automatic diagnosis,” in *Proc. Mtg. North America Fuzzy Information Process. Syst.*, Pensacola, FL, 1998.

[28] ——, “A diagnosis methodology for continuous time measurements using hierarchical signal representations,” in *Proc. IEEE Int. Conf. SMC 3*, San Diego, CA, 1998.

[29] G. A. Carpenter and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Comput. Vis. Graph., Image Process.*, vol. 37, pp. 54–115, 1987.

[30] M. B. Tümer, "A fuzzy syntactic approach to fault diagnostics by analysis of time sampled signals," Ph.D. dissertation, Marquette Univ., Milwaukee, WI, 1998.

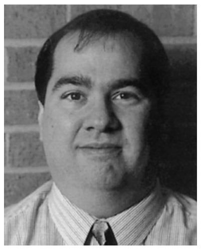
[31] University of Stuttgart, *Stuttgart Neural Network Simulator User Manual Version 4.0*, 1995.

[32] H. J. L. Marriott, *Practical Electrocardiography*. Baltimore, MD: Williams and Wilkins, 1983.

M. Borahan Tümer (M'86) received the B.S. and M.S. degrees, both in computer engineering, from Bogazigi University, Istanbul, Turkey, in 1987 and from Istanbul Technical University, Istanbul, Turkey, in 1990, respectively. He received the Ph.D. degree in electrical and computer engineering from Marquette University. Milwaukee, WI, in 1998.

He has been an Assistant Professor in the Computer Engineering Department at Marmara University, Istanbul, Turkey, since 1999. His current research interests are in learning systems, learning automata, adaptive medical signal processing, neural networks, and fuzzy systems.

Dr. Tümer has been a Member of the IEEE Systems, Man, and Cybernetics Society since 1995.



Lee A. Belfore, II (M'89–SM'00) received the B.S. degree in electrical engineering from Virginia Technology, Virginia, in 1982, the M.S.E. degree in electrical engineering and computer science from Princeton University. Princeton, NJ, in 1983, and the Ph.D. degree in electrical engineering from the University of Virginia, Charlottesville in 1990.

From 1982 to 1985, he was a Member of Technical Staff at AT&T Information Systems. From 1987 to 1988, he was a research scientist in the Department of Electrical Engineering, Center for Semicustom Integrated Systems at the University of Virginia, Charlottesville,Virginia. From 1990 to 1997, he was with the Department of Electrical and Computer Engineering at Marquette University, Milwaukee, WI. Since 1997, he has been with the Department of Electrical and Computer Engineering at Old Dominion University, Norfolk, VA. His research interests include neural networks, data compression, and internet based interactive virtual reality applications.

Dr. Belfore is a member of both the Computer and Systems, Man, and Cybernetics Societies, and the IEEE.

Kristina M. Ropella (SM'xx) **AU: PROVIDE YEARS OF MEMBERSHIP.** received the B.S. degree in biomedical engineering from Marquette University, Milwaukee, WI, in 1985 and the M.S. and Ph.D. degrees, both in biomedical engineering, from Northwestern University, Evanston, IL, in 1987 and 1989, respectively.

She is currently an Associate Professor of Biomedical Engineering at Marquette University, Milwaukee, WI, where she has been a member of the faculty since 1990. Her research interests are in physiologic signal processing, electrophysiology of heart and brain, cardiac arrhythmias, and functional magnetic resonance imaging. She teaches undergraduate and graduate courses in biomedical signal processing, statistical time series analysis, biomedical computing, biomedical instrumentation design and modeling of dynamic systems. She directs a joint doctoral degree program in Functional Imaging offered by Marquette University and the Medical College of Wisconsin and she also codirects the cooperative education program in biomedical engineering.

Dr. Ropella is a Senior Member of the IEEE Engineering in Medicine and Biology Society, and has served on the Administrative Committee. She is a past recipient of the Marquette University Robert and Mary Gettel Faculty Award for Teaching Excellence. She currently serves on the Board of Directors for the Biomedical Engineering Society Board. She is also a member of the North American Society for Pacing and Electrophysiology, the International Society for Computerized Electrocardiology, the American Society for Engineering Education, Sigma Xi and Tau Beta Pi.