2008 International Symposium on Nonlinear Theory and its Applications
NOLTA'08, Budapest, Hungary, September 7-10, 2008

# TS-Type Fuzzy Automaton for Supervisory Control

Janos L. Grantner† and George A. Fodor‡

† Dept. of Electrical and Computer Engineering, Western Michigan University
Kalamazoo, MI 49008-5329, U.S.A.
‡ABB AB
SE-72159, Vasteras, Sweden
Email: janos.grantner@wmich.edu, george.a.fodor@se.abb.com

**Abstract–** Event-driven, large control systems often encounter unexpected events in an uncertain environment. Using a fuzzy automaton offers an effective approximation method to model continuous and discrete signals in a single theoretical framework. A Max-Min automaton can successfully model a cluster of relevant states when a decision is to be made on the next state of a goal path at the supervisory level. However, to provide analytical proof for stability of a fuzzy controller a Takagi-Sugeno inference model is preferred. In this paper a TS-type fuzzy automaton is proposed.

## 1. Introduction

Among the problems that characterize industrial process control innovation, and which are not domain-related, some of the difficult ones are as follows: (a) how can new knowledge be introduced into a system, (b) how can the system activate stored domain knowledge in an autonomous way, (c) how can the knowledge be validated (or otherwise detected as inappropriate) and (d) how can the system recover if the new, activated knowledge (or the currently active knowledge) is not suitable to handle the situation at hand.

With respect to problem (d): there is a need for computationally inexpensive fault detection and identification (FDI) algorithms, and automated recovery from faults. One aspect of FDI and automated recovery from faults is the evaluation of the state transitions between states of a large, complex system. It can be accomplished by focusing only on clusters of relevant states along the goal path to find a suitable next state. A reconfigurable virtual Max-Min fuzzy automaton (also referred to as the Hybrid Fuzzy-Boolean Finite State Machine, HFB-FSM) can be used to model those clusters of states. The other aspect of recovery is devising actuator values in the chosen new state that facilitate the recovery while keeping the system stable. A TS-type fuzzy automaton is proposed to achieve this goal.

The rest of the paper is organized as follows: in Section 2 the key properties of the HFB-FSM model are summarized. In Section 3 the definition of the proposed TS-type fuzzy automaton model (TSTFA) that has been developed from the HFB-FSM is given. In Section 4 the concept of the virtual TSTFA automaton is described. Conclusions are given in Section 5.

## 2. Extended HFB-FSM Automaton Model

The model of the Hybrid Fuzzy-Boolean Finite State machine was presented in [1]. It was extended in [2] to address the modeling requirements of a complex hybrid system at a more flexible level. The notion of this fuzzy automaton is based upon the premises as follows: the fuzzy automaton can stay in some crisp states simultaneously, to a certain degree in each. Those degrees are defined by a state membership function. For each fuzzy state there is just one dominant (crisp) state, though, for which the state membership is a 1 (full membership). Each dominant state is associated with a linguistic model for inference. For each fuzzy state a composite linguistic model is devised using the composition of the linguistic models of those contributing crisp states that has a greater than 0 state membership degree in that fuzzy state. The transitions between fuzzy states are based upon the transitions defined between their dominant crisp states.

There is an underlying Boolean finite state machine to implement the fuzzy automaton. The states of this Boolean automaton are the dominant crisp states of the fuzzy automaton. The fuzzy inputs (and even fuzzy outputs, as an option) are mapped to sets of two-valued logic variables using the B algorithm [1]. The analog inputs with threshold also yield two-valued logic variables. In addition, the automaton may have two-valued inputs as well. All of these Boolean variables are used to devise the next states of the two-valued state variables of the underlying Boolean automaton. Two-valued outputs are devised using all types of inputs and the current dominant state. Fuzzy outputs are obtained through the compositional rule of inference. Defuzzified outputs are calculated by using a suitable defuzzification algorithm.

Formally, a HFB-FSM automaton with p states is defined by the set of equations below:

$$S_{Fk} : S_k , g_{Sk} \tag{1}$$

$$R^* = f(G, R_S) \tag{2}$$

$$G = \begin{bmatrix} \beta_1^1 \ldots \beta_p^1 \\ \ldots\ldots \\ \beta_1^k \ldots \beta_p^k \\ \ldots\ldots \\ \beta_1^p \ldots \beta_p^p \end{bmatrix} \quad (3)$$

$$Z_F = X_F \circ R^* \quad (4)$$

$$Z_C = DF(Z_F) \quad (5)$$

$$X_T \; is \; TRUE \; if \; X_A \geq X_{AT} \quad (6)$$

$$X_B = B(X_F) \quad (7)$$

$$Z_B = B(Z_F) \quad (8)$$

$$Y_B = f_y(X_B, W_B, X_T, Z_B, y_B) \quad (9)$$

$$U_B = f_u(X_B, W_B, X_T, y_B) \quad (10)$$

In Equation (1) a fuzzy state is defined by a dominant crisp (Boolean) state along with a state membership function: $S_{Fk}$ stands for fuzzy state k, $S_k$ represents crisp state k, and $g_{Sk}$ is the state membership function associated with $S_k$ (k=1,…, p). G stands for the matrix of state membership functions. $X_F$, $W_B$, and $X_A$ stand for fuzzy, two-valued (Boolean) and analog inputs with associated $X_{AT}$ threshold values, respectively. A threshold comparator module compares the value of each analog signal with its associated threshold value to set the corresponding $X_T$ signal as true, or false. $Z_F$, $Z_C$, and $U_B$ stand for fuzzy, crisp (defuzzified), and two-valued (Boolean) outputs, respectively.

In Equation (2) R* is the composite linguistic model, and ∘ is the operator of composition in Equation (4). Each crisp state of the HFB-FSM is characterized by an aggregated, overall linguistic model, $R_S$ or by a set of linguistic sub-models in the case of multiple-input-single-output (MISO), and multiple-input-multiple-output (MIMO) systems. For each fuzzy state of the HFB-FSM model, a R* composite linguistic model is created from the finite set of $R_{Si}$ overall linguistic models (i=1,..,p).

Let the HFB-FSM be in fuzzy state $S_{Fk}$, then

$$R_k^* = \max[\min(\beta_1^k, R_{S1}), ..., \min(\beta_p^k, R_{Sp})] \quad (11)$$

where $\beta^k_1$,…, $\beta^k_p$ stand for the degrees of state membership function $g_{Sk}$ and $R_{S1}$,…, $R_{Sp}$ are the aggregated linguistic models [1] in crisp states $S_1$,…, $S_p$, respectively. In fact, Equation (11) is the computational algorithm for Equation (2). By modifying the β degrees of the state membership functions on-line, new R* composite linguistic models can be created under real-time conditions. The R* composite linguistic model is one of the key concepts in defining the HFB-FSM automaton: it decides how fuzzy outputs are inferred from the knowledge base in different fuzzy states. In other words, it reflects the fact of a fuzzy state transition by inferring different fuzzy outputs even for identical fuzzy inputs in two different states.

$X_B$, $Z_B$, $Y_B$, and $y_B$ stand for two-valued Boolean inputs, Boolean outputs (both devised from input and output fuzzy sets using the B algorithm) and next states and present states of the state variables, respectively. The $Z_C$ crisp values of the fuzzy outputs are obtained by evaluating a defuzzification strategy, DF.

The transitions between active composite linguistic models are determined by the state transitions of the HFB-FSM. The state transitions of the HFB-FSM are specified by means of a sequence of changes in the states of the fuzzy inputs (optionally, fuzzy outputs, too), of the analog inputs with threshold, as well as of the two-valued inputs. The changes in the states of the fuzzy inputs are mapped into a corresponding sequence of changes of Boolean input variable sets using the B algorithm. In this two-valued domain, those changes are joined by the state changes of the two-valued inputs and the true/false logic values of the analog inputs with threshold. This combined Boolean input sequence specification is used to synthesize the crisp finite state machine section of the HFB-FSM. Hence, the HFB-FSM model allows the integration of fuzzy, analog and two-valued logic specifications to describe a system's dynamic behavior.

The integrated treatment of fuzzy, analog with threshold, and two-valued signals is of great importance for designing complex hybrid systems.

### 3. TS-Type Fuzzy Automaton Model (TSTFA)

The TSTFA model has been developed from the HFB-FSM one such that the computational algorithm for the linguistic model of Equation (2) and the inference algorithm of Equation (4) are replaced by new equations to comply with the Takagi-Sugeno (TS) model of fuzzy systems [3]. Equation (5) is dropped because there is no need for it in a TS system. Equation (8) is also dropped because no fuzzy output is inferred from the linguistic model anymore. Equations (1), (3) and (6)-(10) remain in effect, however, Equation (9) is slightly revised due to the drop of $Z_B$.

In the TS model [3] the format of implications is proposed as follows:

$$R: If \; x_1 \; is \; A_1, ..., x_k \; is \; A_k \; then \; y = g(x_1, ..., x_k)$$

and g is a linear function such that

$$g = p_0 + p_1 x_1 +, ..., p_k x_k \quad (12)$$

where $A_1$,…, $A_k$ are fuzzy sets, $x_1$,…, $x_k$ are fuzzy inputs and output y is obtained as a crisp value. Suppose there are n implications $R^i$ (i=1,…, n) of the above format. When the fuzzy inputs are given as singletons

$$x_1 = x_1^0, ..., x_k = x_k^0$$

then the final output y is inferred in the following steps.

1) For each implication $R^i$ (I = 1,…, n) $y^i$ is calculated by the function $g^i$ in the consequence

$$y^i = p_0^i + p_1^i x_1^0, ..., + p_k^i x_k^0 \quad (13).$$

2) The truth value of the proposition y=y$^i$ is calculated by the equation

$$| y = y^i | = (A_1^i(x_1^0) \wedge ... A_k^i(x_k^o)) \wedge | R^i | \quad (14)$$

where |*| means the truth value of proposition *, $\wedge$ stands for min operation and A(x$^0$) stands for the grade of membership of x$^0$ in fuzzy set A. For simplicity, |R$^i$| = 1is assumed.

3) The final output y inferred from n implications is given as the average of all y$^i$ with the weights |y = y$^i$|:

$$y = \frac{\sum | y = y^i | \times y^i}{\sum | y = y^i |} \quad (15)$$

In the TSTFA model the Takagi-Sugeno version of the IF THEN rules is adopted as it is given in Equation (12). The only change in the notation is that Z is used for output rather than y.

$$R: If\ x_1\ is\ A_1, ..., x_k\ is\ A_k\ then\ Z = g(x_1, ..., x_k)$$

and g is a linear function such that

$$Z = p_0 + p_1 x_1 +, ..., p_k x_k \quad (16)$$

In each crisp state S$_k$ the final output Z$_{Sk}$ is calculated according to Equations (14) and (15) above:

$$| Z_{Sk} = Z_{Sk}^i | = (A_1^i(x_1^0) \wedge ... A_k^i(x_k^o)) \wedge | R^i | \quad (17)$$

$$Z_{Sk} = \frac{\sum | Z_{Sk} = Z_{Sk}^i | \times Z_{Sk}^i}{\sum | Z_{Sk} = Z_{Sk}^i |} \quad (18)$$

The notion of composite output Z$^*$ is introduced to reflect the contribution of the output values devised from the TS linguistic models that are attached to crisp states to the final output in a fuzzy state. Let the TSTFA be in fuzzy state S$_{Fk}$, then

$$Z_k^* = \frac{\beta_1^k Z_{S1} + ... + \beta_p^k Z_{Sp}}{\sum_i \beta_i^k} \quad (i = 1, ..., p) \quad (19)$$

It is clear from Equation (19) that only those crisp states that have greater than 0 degree of state membership in fuzzy state S$_{Fk}$ contribute to the final output.

Formally, a TSTFA automaton with p states is defined by the following set of equations:

$$S_{Fk} : S_k, g_{Sk} \quad (20)$$

$$R_S = TS(X_F, Z_S) \quad (21)$$

$$G = \begin{bmatrix} \beta_1^1 ... \beta_p^1 \\ ..... \\ \beta_1^k ... \beta_p^k \\ ..... \\ \beta_1^p ... \beta_p^p \end{bmatrix} \quad (22)$$

$$Z^* = TS(X_F^0, R_S, G) \quad (23)$$

$$X_T\ is\ TRUE\ if\ X_A \geq X_{AT} \quad (24)$$

$$X_B = B(X_F^0) \quad (25)$$

$$Y_B = f_y(X_B, W_B, X_T, y_B) \quad (26)$$

$$U_B = f_u(X_B, W_B, X_T, y_B) \quad (27)$$

TS stands for a reference to the Takagi-Sugeno model. The detailed representation of Equation (21) is given by Equation (16). The computational algorithm for Equation (23) is given by Equations (17)–(19). X$_F^0$ stands for singleton fuzzy inputs. The block diagram of the TSTFA automaton is shown in Fig. 1.

One can notice that the B algorithm that maps changes in a fuzzy input to changes of the states of a set of Boolean input variables will be executed much faster for the TSTFA than for the HFB-FSM. It is due to the fact that there is no need to run a defuzzification step if the fuzzy inputs are singletons.

## 4. Virtual TSTFA Automata for Supervisory Control

Most systems that are being used in industry contain a large amount of knowledge including fault detection, isolation and recovery (FDIR) algorithms. Contemporary control systems employ the so-called agent architectures [4]. A software agent can gather information about its environment and can perform actions which change some internal state, or representation of the agent such that either a numerical, or a declared qualitative, or a behavioral optimum is achieved. Our hypothesis is that a real intelligent agent behavior requires a dynamic architecture that can combine overall agent population goal changes with architecture reconfiguration and with agent model-based behavior. One of the key functions of the agent is fault detection and assessment. If the agent recognizes that a fault cannot be managed via the application, it will assess if the fault is recoverable in principle. A reconfigurable virtual TSTFA automaton will be used to model the actual state of a hybrid system and to accomplish this task.

The concept of the virtual fuzzy automaton was introduced in [2]. A hardware accelerator for reconfigurable fuzzy automata was proposed in [5]. Since the state set of a complex system may consist of tens of thousands of states, it would be impractical to design a fuzzy automaton of that size. However, as it is shown in [6], when the current goal state can be computed and all

possible disturbances are known, only a small partition of the total state space is relevant to make a decision on the next state along a goal path. A supervisory algorithm monitoring the flow of states first determines a segment of the transition graph (a cluster of relevant states) prior to the decision on the next state transition. Then it creates an instance of the virtual TSTFA to implement it. In other words, it defines a TSTFA model of the relevant cluster of states of the goal path.

The following information will be downloaded to the reconfigurable TSTFA: the active sets of fuzzy, analog with threshold, and Boolean inputs and fuzzy and two-valued outputs, respectively, state membership function degrees, the actual mapping scheme between fuzzy and Boolean subintervals of the B algorithm, threshold values, the initial state, the state transition graph along with the conditions for state changes, the mapping function for two-valued outputs, the Takagi-Sugeno linguistic models and then the current status of all active input signals of any kind. The TSTFA will then make a decision on the transition to the next state, infer outputs (if needed), and pass all these information back to the supervisory algorithm. The configuration process will repeat each time when a new instance of the TSTFA should be created to track the current segment of the state graph.

## 5. Conclusions

Dynamic software architecture is a key element that allows a flexible mapping between informational resources and the active components in industrial systems. It can be implemented using software agents instead of traditional communications channels. The software agents, when enhanced with fuzzy automata, can perform a learned, model-based reconfiguration that optimizes what type of system behavior has priority in the current situation. A Takagi-Sugeno type fuzzy automaton model is introduced to support decision making with respect to automatic recovery from faults.

## References


[1] J.L. Grantner, G. Fodor, D. Driankov, "Hybrid Fuzzy-Boolean Automata for Ontological Controllers", Proceedings of the 1998 IEEE World Congress on Computational Intelligence, FUZZ-IEEE'98, Vol. I, pp. 400-404, Anchorage, Alaska, May 4-9, 1998.

[2] J. L. Grantner, G. A. Fodor, D. Driankov, "The Virtual Fuzzy State Machine Approach - A Domain-Independent Fault Detection and Recovery Method for Object-based Control Systems", 18th International Conference of the North American Fuzzy Information Processing Society - NAFIPS'99, June 10-12, 1999, New York, NY, Proceedings, pp. 158-162.

[3] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application in modeling and control," *IEEE Trans.System, Manand Cybernetics,* vol. 15, 1985, pp. 116-132

[4] M. Knapik, J. Johnson, "*Developing Intelligent Agents for Distributed Systems, Exploring Archtiecture, Technologies and Applications*", McGraw-Hill, 1998.

[5] J. L. Grantner, P. A. Tamayo, R. Gottipati and G. A. Fodor, "Reconfigurable Fuzzy Automaton for Software Agents" , Proceedings of the BISCSE 2005 Conference, November 2-5, 2005, University of California, Berkeley, Berkeley, CA, on CD

[6] G. A. Fodor, "*Ontologically Controlled Autonomous Systems: Principles, Operations and Architecture*", Kluwer Academic Publishers, Boston / Dordrecht / London 1998


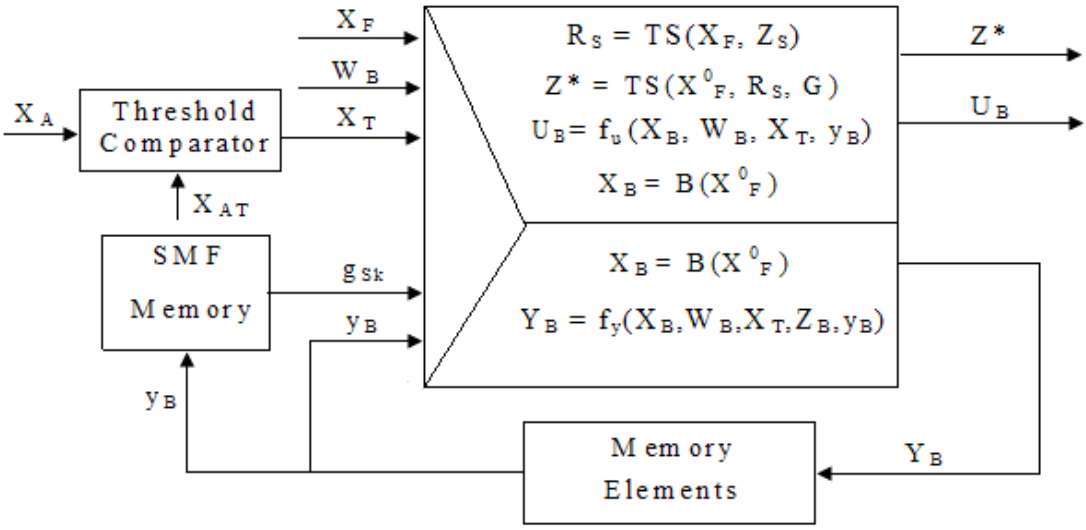


Figure 1. TSTFA Automaton Block Diagram