Threshold Fuzzy Systems:
A Priority-based Hierarchical Control Scheme

James L. Overholt
National Automotive Center
U.S. Army Tank-automotive & Armaments Command
Warren, MI 48397-5000

K. C. Cheok, G. Edzko Smid
Department of Electrical and Systems Engineering
School of Engineering and Computer Science
Oakland University, Rochester MI 48309-4401

ABSTRACT

In this paper we will introduce a new trainable control methodology called Threshold Fuzzy Systems (TFS). The need for TFS methods comes from controlling plants that exhibit a *varying priority hierarchy*. A trademark of these variable priority systems is that *if a lower priority system task is completed before a higher priority one the system could become unstable*. Designing a control structure that will take into account the varying priorities of a system is difficult. TFS control will be shown to be a good approach to these types of systems. Key to the proposed TFS structure will be the inclusion of a Behaviorist Fuzzy Rulebase (BFR) and a Rule Dominance Mechanism (RDM) into the classical fuzzy control architecture. The purpose and uses of the BFR and RDM will be explained. The TFS control scheme will be presented and symbolic and numerical examples will be presented.

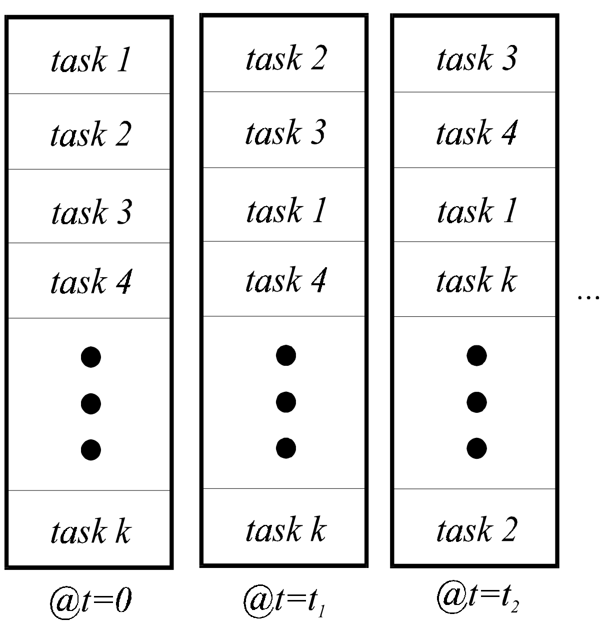

Figure 1: Task Priority Listings

INTRODUCTION

Hierarchical systems contain multiple objectives that need to be satisfied in order for the overall system to perform its designed function. These tasks will each have an associated priority, which is used to properly place the emphasis of the system control. The control policy created for this type of hierarchical structure must strive to accomplish all of the tasks according to the order dictated by the relative priorities. In actuality, the priority listing of each task is not fixed but will change based on factors such as the current state(s) of the plant and the magnitude of the control (called a *varying or "dynamic" priority hierarchy*).

In addition, these priority changes will occur with very little (if any) forewarning (**Figure 1**). Perhaps the biggest concern is that with the shifting priorities of the system, if a lower priority task is accomplished it may result in the system becoming unstable. Hierarchical systems control policies need to incorporate a structure, which tries to anticipate and handle priority changes in the multiple tasks of the system (**Figure 2**).

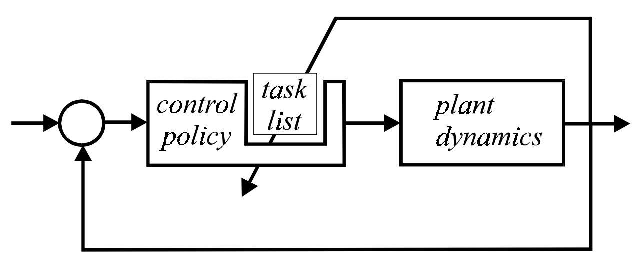

Figure 2: Varying Priority Control

THRESHOLD FUZZY SYSTEM

We introduce a prioritization-based control approach called *Threshold Fuzzy Systems* (TFS). Threshold Fuzzy Systems are defined as follows;

A **Threshold Fuzzy System** will include all the standard components of a Mamdani fuzzy system [1] with the additional feature of a **Behaviorist Fuzzy Rulebase** (BFR) as well as the inclusion of a **Rule Dominance Mechanism** (RDM). Behaviorist fuzzy rules are a set of simply structured rules used for the control of a single task. The **RDM** is used to modify the outputs of each individual fuzzy rule, based on the outputs of other *conflicting fuzzy rules* achieving some pre-determined threshold.

Figure 3 shows the components of a TFS. **X** is the crisp inputs into the system, **Y** is the crisp outputs of the system, **C** is the contribution of each individual rule and **D** is the dominated contribution of each rule. The *mathematical* definition of rule contribution is based on the inferencing technique used in the fuzzy system.

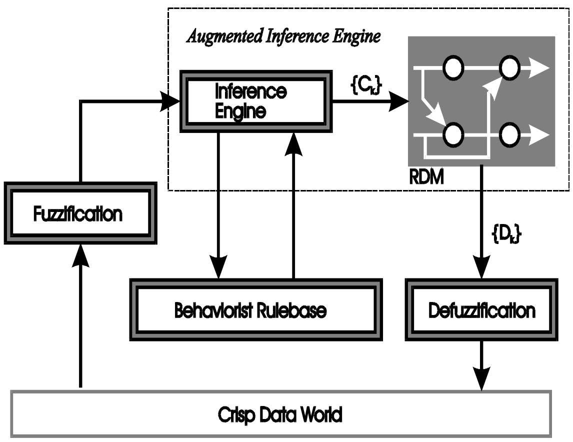

Figure 3: Threshold Fuzzy System Components

Given the behaviorist fuzzy rule **R: If x is A then y is B** and an occurrence of **x = <u>x</u>,** then the *contribution* of rule '**R**' is defined as;

$$C_R = \int \left[\mu_A(\underline{x}) * \mu_B(y)\right] * dy; \text{ for 'product' inferencing [2].}$$

Hierarchical plants are made up of a series of tasks that each has a relative priority. One approach to controlling this type of system is to design a TFS complete with a BFR and an RDM structure. Each rule of the BFR should be governed by the following criteria;

1. a behaviorist rule is a mapping from a single input stimulus to a single output control;
2. each mode of stimulus corresponds to a dimension of the input space and is independent of other stimulus modes; and
3. triggering of a behavior takes place when the current input data and the antecedents of a behaviorist rule have a non-empty fuzzy intersection.

These guidelines will lead to rules having **single antecedents** and **single consequences** (although in some special cases more then one antecedent per rule would be acceptable). A set of behaviorist rules can be constructed to handle the control of several loosely coupled tasks. Several of these task-based rulebases will constitute the formation of a *behaviorist fuzzy rulebase*. This approach is consistent with [3] and [4] in behavioral approaches to complex systems control.

FUZZY RULE CONFLICT DEFINITION

With the single input/single output structure of the BFR, many of the rules will conflict and have a tendency to cancel each other out. In order to use an RDM to account for the conflicting nature of these rules it is first necessary to define a formal definition of fuzzy conflict.

Given the two behaviorist rules; **If X_1 is A then Y is S** and **If X_2 is B then Y is T**, then the rules are said to be in ***Fuzzy Conflict*** if all of the following 3 conditions hold;

1. the **Output Variables** are identical,
2. the **Output Fuzzy Sets** [2] are *not* identical (i.e. **S** ≠ **T**), and
3. the **Contributions** of each rule are both non-zero (i.e. $\mathbf{C_R} > 0$).

If two rules have the same control action variable, but have different linguistic consequences, they are in fuzzy conflict. For example, the following two rules are said to be in fuzzy conflict because their output variables are the same (**FORCE**) and their consequences (fuzzy set labels) are different (**LARGE POSITIVE** and **LARGE NEGATIVE**).

Rule 1:	If **CART_POSITION** is **LARGE LEFT** then **FORCE** is **LARGE POSITIVE**
Rule 2:	If **CART_VELOCITY** is **POSITIVE** then **FORCE** is **LARGE NEGATIVE**

RDM DEFINITION

The inclusion of a BFR into the standard *Mamdani fuzzy control* scheme is not sufficient to control all of the system responses of *multi-objective plants*. The addition of an RDM acts to modify the *contribution* of a fuzzy rule if a conflicting rule's *contribution* has reached a predetermined level. The addition of an RDM also serves to augment the *fuzzy inferencing process*. **Figure 4** shows the general mechanics of the process.

The crisp inputs into the system interact with the rulebase and inference engine to produce contributions for every rule. Every rule contribution can be further modified by the RDM. Whether further changes take place depends on the dominance function F_i and the contribution levels of the other rules in the rulebase. **Figure 5** shows the i^{th} node of the RDM.

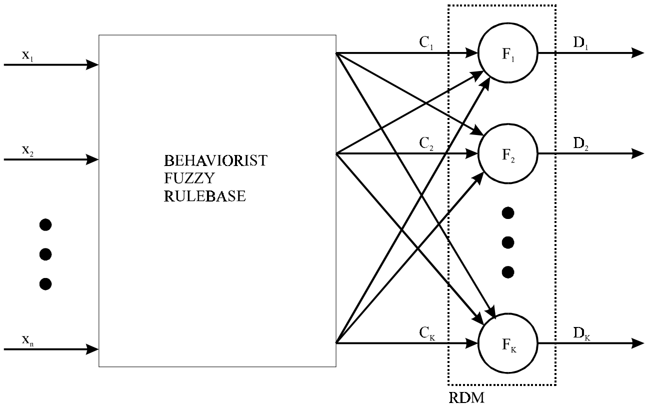

Figure 4: K-Rule Behaviorist Fuzzy Rulebase with a Companion RDM

$\bullet_i$ are the contributions of those rules which conflict with rule '**i**'. $\mathbf{F_i}$ is the dominance function, which modifies $\mathbf{C_i}$ into its final form, $\mathbf{\{D_i\}}$. The final set of *dominated* rule contributions (**D**) are combined in a standard defuzzification scheme. This represents the conflict resolution capability of the RDM. Within this RDM definition, there are many possible implementations of the RDM that will act to modify the contributions of the BFR. One such implementation is called a **Cascading RDM**.

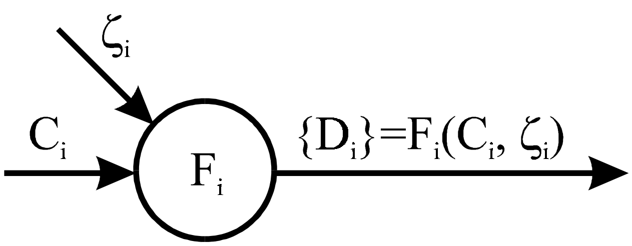

Figure 5: Rule Contribution Modification

CASCADING RULE DOMINANCE MECHANISM

A *Cascading RDM* is an adaptive node network [5] where each node has 2 inputs; one is the contribution that is being *dominated* and the second is the *dominating* contribution. This corresponds to a conflicting pair of behaviorist fuzzy rules. Each rule contribution is altered as it passes through a series of *nodes* ($\mathbf{n_{i,j}}$) along its individual *contribution path*. The level of modification at each node is dictated by a *dominance function* ($\mathbf{F_{i,j}}$). Each individual dominance function will use the conflicting rule *normalized* contribution as its independent input variable. *Normalized* refers to the actual rule contribution as a percentage of the maximum possible rule contribution. This will restrict the domain of the dominance function to lie between **(0,1)**. **Figure 6** shows the dominance function and how it uses the conflicting rule contribution to generate a scaling variable $\mathbf{F_{i,j}}(\mathbf{C_j})$ that modifies the input contribution. The value of the dominance function is then used to attenuate the input contribution by using either a 'min' or 'product' inferencing method [2]. The inferencing method should correspond to the inferencing scheme used in the BFS, i.e.

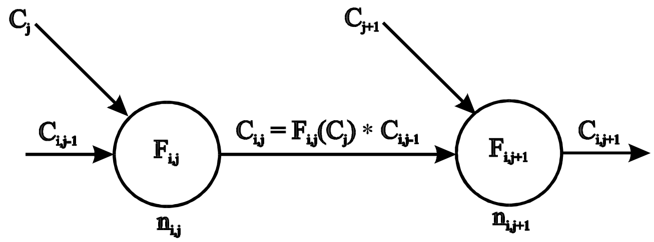

Figure 6: Cascading RDM Contribution Path

'product' inferencing method $\rightarrow \mathbf{F_{i,j}(C_j) \circ C_{i,j-1} \rightarrow C_{i,j} = F_{i,j}(C_j) * C_{i,j-1}}$, or

'min' inferencing method $\rightarrow \mathbf{F_{i,j}(C_j) \circ C_{i,j-1} \rightarrow C_{i,j} = \min\{F_{i,j}(C_j), C_{i,j-1}\}}$.

Once the rule contribution has passed through all of the nodes along its contribution path, the final node output serves as the final dominated rule contribution (i.e. if the BFS has 'k' rules, then $\mathbf{C_{i,k}} \equiv \mathbf{D_i}$).

GEOMETRIC INTERPRETATION

For this example, all 3 behaviorist fuzzy rules "conflict" with each other (*mutually conflicting*). **Figure 7** shows a geometric interpretation of the 3-rule cascading RDM. The contributions of the 3-rule BFS are on the left side of the figure. Each contribution is fed into the cascading RDM along its own path. Contributions are modified at each node until a dominated set of contributions emerge at the other side. Note that certain nodes in the RDM (i.e. $\mathbf{n_{i,i}}$) have no *dominating* input. These nodes correspond to where a rule would conflict with itself. By the definition of fuzzy conflict this could not happen. In general, *behaviorist fuzzy rules do not conflict with themselves*. These nodes have a '**1**' inside.

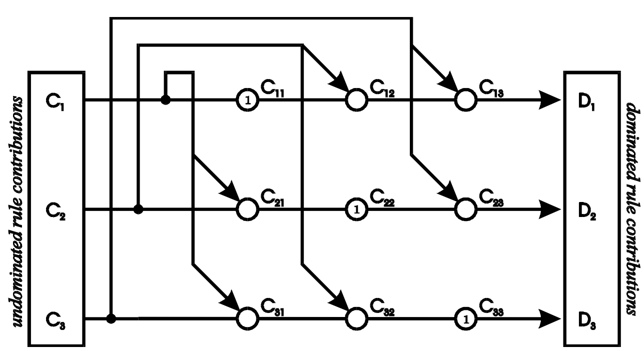

Figure 7: Cascading Rule Dominance Mechanism

Figure 8 presents a geometric example of a mutually conflicting 3-Rule system. Directly beneath the 'undominated' contributions is a graphic illustrating the results of a center average defuzzification (CAD) scheme [2] for the BFS without an RDM. Directly beneath the 'dominated' contributions is a graphic depicting the results of a CAD calculation for the BFS with an RDM.

Comparing the results show that the output shifted to the right for the 'dominated' set of contributions. The contribution of rule #2 was affected the least. The contribution of rule #1 was entirely eliminated by the domination of rule #3. Rule #3 was reduced solely by the contribution of rule #2. The end result shows that the system was modified in favor of rule #2.

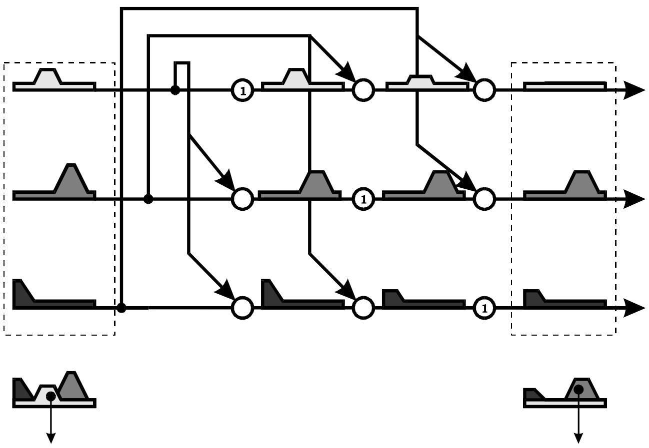

Figure 8: Mutually Conflicting 3-Rule System

The geometric representation of a 2-rule system is given in **Figure 9**. The example dominance functions f_{12} (i.e. rule 2 'dominates' rule 1) and f_{21} (i.e. rule 1 'dominates' rule 2) are as follows;

$$f_{12}(\hat{C}_2) = \frac{1}{1+e^{15(\hat{C}_2-0.4)}}; \quad f_{21}(\hat{C}_1) = \frac{1}{1+e^{25(\hat{C}_1-0.8)}},$$

where $\hat{C}_1$ and $\hat{C}_2$ are the normalized contributions of rules 1 and 2, respectively.

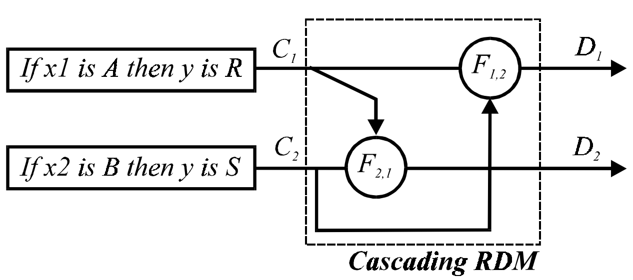

Figure 9: 2-Rule BFS for Cart Centering

Using the TFS approach, including product inferencing and a CAD strategy, the control maps for this 2-rule system (with and without an RDM) are generated (**Figure 10**).

TFS AS AN ADAPTIVE NODE NETWORK

The TFS structure has many advantages over feedforward artificial neural networks (ANN). Specifically, the connection topology of the RDM is uniquely defined once the behaviorist rulebase is given. Also, with the addition of new rulebases, training only has to be conducted on the dominance functions associated with the new rules where there is a conflict with the original rulebase. This leads to modularity for Threshold Fuzzy Systems that supercedes ANNs.

TRAINING OF THE DOMINANCE FUNCTIONS

The most commonly used dominance function will be the sigmoid function, described as

$$f_{i,j}(C_j) = \frac{1}{1+e^{-m_{i,j}(\hat{C}_j - b_{i,j})}},$$

where $f_{i,j}$ is the j^th dominance function on the i^th contribution path, Cj is the j^th rule contribution, $\hat{C}_j$ is the normalized contribution of the j^th rule, $m_{i,j}$ is the slope of the sigmoid and $b_{i,j}$ is the mid-point locator of the function. Using input-output training data, the slope ($m_{i,j}$) and center ($b_{i,j}$) of each dominance function can be iteratively found.

The update equation for $m_{i,j}$ is given as…

$$\left[m_{i,j}\right]_{new} = \left[m_{i,j}\right]_{old} - \alpha \cdot \Phi_i^p \cdot \left[\mu^j(x^p) - b_{i,j}\right],$$

and the update equation for $b_{i,j}$ is given as…

$$\left[b_{i,j}\right]_{new} = \left[b_{i,j}\right]_{old} + \beta \cdot \Phi_i^p \cdot m_{i,j},$$

and Φ_i^p is described as…

$$\Phi_i^p = C_i \cdot D_i^p \cdot \left[\hat{y}(x^p) - y^p\right]\left[1 - f_{i,j}(C_j)\right]\left[\frac{\rho_i v^p - u^p}{(v^p)^2}\right],$$

where,

α and β = learning rates,

$\hat{y}(x^p)$ = *TFS* output for *p*,

y^p = target output for *p*,

C_i = contribution of the i^th rule,

D_i^p = the i^th contribution path's output for *p*,

$\mu^j(x^p)$ = fuzzifcation of the crisp input of the j^th rule,

u^p = numerator of the CAD scheme, and

v^p = denominator of the CAD scheme.

These equations can be used to update the parameters of the dominance functions called out in the RDM.

AUTONOMOUS NAVIGATION EXAMPLE

The TFS control methodology is has been demonstrated on a motorized wheel chair model of an autonomous vehicle system (AVS). The AVS must find a beacon in an unknown environment. The only external sensory inputs will be 6 sonar sensors and a directional beacon detector. For this simulation exercise these sonars will be able to accurately sense the distance of the closest object in it's field of view (~30 degrees). The directional beacon sensor will be able to determine the direction of the target within ±10% of the actual heading angle. The distance to the target will remain unknown.

A set of behaviorist fuzzy rules was developed to quickly acquire a target under the sensory input described above. An RDM was developed such that obvious conflicting situations could be handled (such as conflicts that occur when the beacon directional finder points north but a north facing sonar sensor indicates that an obstacle is also in the same locale).

For the simulations, both a BFR and a RDM were used. In both simulations, the AVS run would end if the vehicle ever got within 3' of an obstacle. In order to terminate the simulation successfully, the AVS had to come within 1' of the final target. If either the target was reached or an obstacle was "touched", the simulation would cease after 180 seconds. In over 300 randomly generated scenarios, the AVS with the RDM outperformed the AVS without the RDM over 75% of the time.

CONCLUSIONS

We have defined the basic concepts of a new fuzzy control architecture called *threshold fuzzy systems*. In addition, we have also developed definitions for *fuzzy conflict* and for the *rule dominance mechanism*. Finally, the usefulness of this new methodology will be most obvious when applying it to the control of multi-objective systems, such as autonomous vehicle navigation. Since it fits into the definition of adaptive-networks, supervised training algorithms.

REFERENCES

1. Mamdani, E. H. "Applications of fuzzy algorithms for control of simple dynamic plant." *Proc. IEE*, 121 (12), 1585-1588.
2. Wang, L. X. *A Course In Fuzzy Systems And Control*. Prentice-Hall, NJ (1997).
3. Brooks, R. A. "A robust layered control system for a mobile robot." *IEEE J. of Robotics Automat*. (March 1986), 14-23.
4. Anderson, T. L. & Donath, M. "Animal behavior as a paradigm for developing robot autonomy." *Robotics and Autonomous Systems* 6 (1990), 145-168.
5. Jang, J.-S. Roger "ANFIS: Adaptive-network-based fuzzy inference system." IEEE Trans. On Systems, Man, and Cybernetics 23 (3), 665-685.

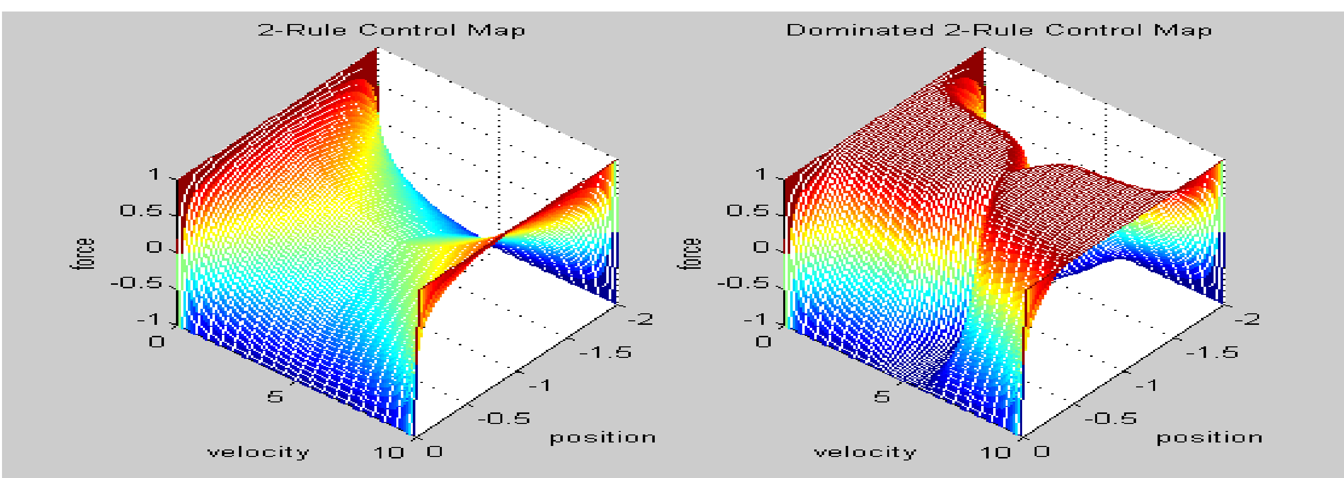

Figure 10: 2-Rule Control Map with and without Cascading RDM