Proceedings of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Singleton-Based Two-String Inference in Recurrent Fuzzy Systems

Moritz Schneider, Jürgen Adamy

Institute of Automatic Control and Mechatronics,
Control Methods and Robotics Lab,
Technische Universität Darmstadt, Germany
e-mail: {schneider, jadamy}@rmr.tu-darmstadt.de

Abstract: Two-string fuzzy inference consists of two separate inference mechanisms: One conventional fuzzy inference system that processes recommending rules, as well as a mechanism for processing negative rules, which prevent the system from outputting their associated values when their premise is fulfilled. Two-string inference has valuable applications in pattern recognition and control tasks. We present a method rendering two-string inference applicable and computationally feasible in recurrent fuzzy systems, i.e. Mamdami-type fuzzy inference systems equipped with defuzzified state feedback. We show the efficiency of our approach by means of an illustrative example from biological systems modelling and suggest application areas for recurrent two-string fuzzy systems.

Keywords: Recurrent Fuzzy Systems; Fuzzy Inference; Rule-based Systems; Dynamic modelling; Fuzzy modelling; Fuzzy control.

1. INTRODUCTION

Function approximators are often employed to design controllers, classificators, or system models. When analytical process models are nonexistent or too expensive to obtain, function approximators supply at least approximate models that are often satisfactory for practical applications. Besides neural networks, which are trained by machine learning algorithms from scratch to minimize approximation error, *fuzzy inference systems* allow for an interactive design stage, where human expert knowledge can be combined with automated parameter or structure optimization. Basically, such systems operate as rule-based transition systems on symbols, where the underlying approximation is achieved by linear interpolation between set points to which aforementioned symbols refer. The advantage of fuzzy methodology is that such rule-based transition systems can in principle be directly *interpreted by humans* and thus offer a possibility to create *transparent* systems. System transparency, on the other hand, affords an upper bound on the number of symbols on which the rules are based. This point bears difficulties, as the *universal approximator property* of fuzzy systems was derived under the condition of an unbounded rule base, as was emphasized in Klement et al. [1999].

Moreover, the design stage of fuzzy systems is often not fully interactive, as it is generally difficult to achieve locally limited effects by tuning from hand, since tuning parameters or structure often, due to interlocking effects, results in completely different behavior. These problems can be adressed by trading size for internal complexity through the establishment of more complex inner semantic structure, provided that said structure reflects the way a human expert would think about the system to model. To this end, *two-string fuzzy inference* has been introduced. Conventional fuzzy systems have one inference string which processes rules that can be considered as *production rules*. They are *constructive*, in the sense that they recommend the system to produce a certain output value, given a certain input value. Two-string fuzzy systems are augmented with a second, separate inference string that processes *negative rules*. This inference mechanism can be thought of as destructive, its effect is to prevent the system from outputting a certain output value when a certain input value occurs.

Two different approaches to two-string inference have been developed independently from another around 1995: (i) *Hyperinference and -defuzzification*, presented for example in Kiendl [1997], and (ii) *p/n fuzzy systems*, which have been developed in Branson and Lilly [2001]. While there are technical differences, both approaches rest on a similar conceptual basis. Also, both have been successfully applied to real world applications: p/n fuzzy systems for control of an omnidirectional mobile robot in Lilly [2007] and image classification in Ngyuen and Wu [2008], Hyperinference and -defuzzification for pattern recognition in Guarracino et al. [2013], different control engineering problems in Kiendl [1998] and Krause [1999], and control of an industrial robot in Schwane et al. [1997]. The Hyperinference approach allows for flexible combination of recommendations and warnings through the use of different *veto strategies*, leading to variations of two-string inference that differ in the underlying interpretation of warnings and how they should influence the system output. Therefore, we consider this approach.

Hyperinference and -defuzzification has been designed for the application in static fuzzy systems. However, in the past years there has been growing interest in dynamical fuzzy systems, i.e. systems with internal states that

Original publication in IFAC Proceedings Volumes (now IFAC-Papers online), https://doi.org/10.3182/20140824-6-ZA-1003.01889

are taken into account when producing a new output value. Besides the well known Takagi-Sugeno fuzzy systems, which are not further discussed here for the sake of brevity, *recurrent fuzzy systems* have been developed. Recurrent fuzzy systems achieve dynamic modelling capabilities through a defuzzified state feedback. Since a *Mamdami-type inference scheme* is employed, this system class offers means for transparent and interpretable system design.

Recurrent fuzzy systems (RFS) were introduced independently by Gorrini and Bersini [1994] and Adamy [1995]. The latter mentioned work has been motivated by their characterization as *fuzzy automata*. Therein, RFS have been used as a fault detection system in an industrial steel casting system. In the former mentioned work, RFS have been studied as fuzzy counterparts to *recurrent neural networks*. The system class has then further been developed in Kempf and Adamy [2003] In Adamy and Kempf [2003], the dynamic behavior of discrete-time RFS has been studied and conditions for *chaotic behavior*, as well as for *automatonlike*-ness and *stability* have been derived. Furthermore, based on the original application of RFS as pattern recognition automata, a general method for sequential pattern recognition employing RFS has been developed in Kempf and Adamy [2004]. Later, RFS were independently used for dynamical systems modelling and process fault detection on the basis of a gradient-based learning method in Schwung [2011], and mobile robot control in Surmann and Peters [2000], using a genetic learning algorithm which has been presented in more detail in Surmann and Maniadakis [2001]. In Flemming [2008], the system class was extended to continuous-time models, which have been shown to be well suited for approximating quantitative dynamics of a system.

Since the prevailing application areas of recurrent fuzzy systems and two-string inference, respectively, are largely congruent, equipping this system class with two-string inference could increase its application potential in these areas. Kiendl [1997] mainly considers complex output membership functions, where the actual output value of the system depends on membership function values on a whole interval of possible output values. This, as the author ackknowledges, is suboptimal concerning computational practicabillity. To achieve computational feasibility in recurrent fuzzy systems, *Center-of-Singleton (CoS) defuzzification* is employed, allowing for efficient computation of the output value as a weighted sum. Unfortunately, Hyperdefuzzification does not apply well to singleton output membership functions, as ackknowledged by Schwane et al. [1997]. The main problem is that in any gravity-based defuzzification scheme, a 'best compromise' in form of a medium value with respect to the output membership function is found, which can result in output values which the negative inference string strongly forbids.

The contribution of this paper is twofold: On one hand, we present a method to overcome this problem, therefore rendering singleton-based two-string inference feasible for use in recurrent fuzzy systems. On the other hand, we outline the benefits of augmenting RFS with two-string inference for applications in control engineering, pattern recognition and dynamical systems modelling. This paper is organized as follows: After recapturing the method of Hyperinference and -defuzzification, we recall the basic definitions of recurrent fuzzy systems. Then, the proposed method is described, followed by a simulation example of a biological systems model to illustrate the efficiency of our approach. Eventually, we give a final discussion and present further work.

2. PRELIMINARIES

2.1 Hyperinference and -defuzzification

In standard fuzzy systems, the inference mechanism is considered to work constructive in the sense that the rule

$$R_r^+ : \text{ IF } L_k^u \text{ AND } L_l^x \text{ THEN } L_m^y \tag{1}$$

recommends the output in the conclusion to the extent to which the premise of the rule is satisfied. Negative rules have the form:

$$R_r^- : \text{ IF } L_k^u \text{ AND } L_l^x \text{ THEN } L_m^y \text{ FORBIDDEN.} \tag{2}$$

In the remainder of this paper, we will omit the 'FORBIDDEN' statement, for the label of a rule already indicates whether it is positive or negative.

Three different mechanisms for combining the outputs of both inference strings were proposed: *strong veto*, *weak veto* and *fuzzy veto*. Strong and weak veto inference calculates the resulting output membership values only on the basis of the recommendations, while the inhibitions are employed to nullify the resulting output membership function whenever a certain warning level is exceeded. For every $x \in X$, strong veto means

$$\mu(x) = \begin{cases} \mu^+(x), & \text{if } \mu^-(x) = 0 \\ 0, & \text{else,} \end{cases} \tag{3}$$

while weak veto is defined as

$$\mu(x) = \begin{cases} \mu^+(x), & \text{if } \mu^+(x) \geq \mu^-(x) \\ 0, & \text{else.} \end{cases} \tag{4}$$

Fuzzy veto, however, is based on a gradually destructive inference, where a fuzzy AND-operator is used to combine recommendations with 'not-inhibitions'. The rationale behind this is to reduce the recommendation of an output value according to the strength with which this value is inhibited. Kiendl [1997] studies three different versions of fuzzy veto:

$$\begin{aligned} \mu(x) &= \mu^+(x) \wedge \neg\mu^-(x) \\ &= \begin{cases} \mu^+(x) \cdot (1 - \mu^-(x)) & \text{prod-veto} \\ \min(\mu^+(x), (1 - \mu^-(x))) & \text{min-veto} \\ \max(0, \mu^+(x) - \mu^-(x)) & \text{diff-veto,} \end{cases} \end{aligned} \tag{5}$$

all of which show distinctly different behavior.

In each case, the *support* of the resulting admissible outputs membership function may not be connected on its whole domain. This raises the question, how to produce a reasonable compromise between the recommendations and warnings produced by positive and negative inference, respectively. When using naive CoS-defuzzification, output values in strongly forbidden regions can occur as output values due to the calculation of output by linear interpolation between allowed core positions. An exemplary situation for this problem is depicted in Figure 1.

The method of Hyperdefuzzification considers each connected component of the support of the admissible output

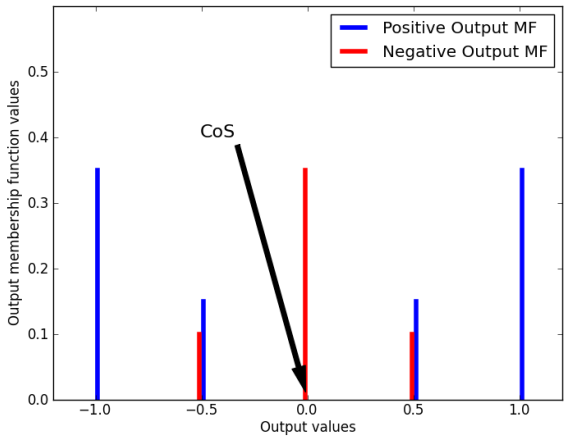

Fig. 1. An example where every Hyperinference strategy leads to an output membership function that can not reasonably be processed by CoS-defuzzification

region separately, e.g. by *Center-of-Gravity* or *Mean-of-Maximum defuzzification*, and outputs a value with respect to some criterion, for example largest-area or maximum-membership-value. This yields considerable computational effort in larger systems. Therefore, Kiendl [1997] also has introduced simplified methods for fast computation, which are based on a generalized defuzzification scheme called *inference filter*. The approach presented in the following section can be conceived as a pure singleton-based alternative to these methods, which we do not discuss in greater detail at the moment due to space restrictions.

2.2 Recurrent Fuzzy Systems

In the following, the definition of RFS as given in Kempf [2004] is revisited. Note that we omit the time-continuous case, for which we refer to Flemming [2008], since extending our approach to this situation is completely straightforward. A recurrent fuzzy system is characterized through the state space equation

$$x(k+1) = f(u(k), x(k)). \tag{6}$$

The function f in (1) consists of a complete fuzzy system with fuzzification, inference and defuzzification. In the case of *multiple input/output* systems, f is a vector function. The algebraic product is used as implication and AND-operator, for the OR-operator the normal sum is used, which allows for efficient computation. In principle, the definition of RFS also encompasses a dedicated output y, which is realized by means of a separate fuzzy system, but since this output does not contribute to the system dynamics and may not even exist in some cases, it will be omitted. Later, we in fact use y as a placeholder-variable to refer to possible system outputs at time k, in contrast to $x(k)$ which is an input variable to the system. This is necessary because the explicit time-dependence of the components of u, x will in further be omitted to save space.

Let the input space $U = [u^{\min}, u^{\max}] \subset R^m$ and the state space $X = [x^{\min}, x^{\max}] \subset R^n$. Then, for each $j \in \{1, \ldots, m\}$, the set $L_{I_j}^{u_j} = \{L_1^{u_j}, \ldots, L_{k_j}^{u_j}\}$ is a finite set of *linguistic values*. Considering the state space, for each $i \in \{1, \ldots, n\}$ there is also a finite set $L_{I_i}^{x_i} = \{L_1^{x_i}, \ldots, L_{k_i}^{x_i}\}$ of linguistic values.

To each linguistic value L^x (L^u), a corresponding *membership function* $\mu_{L^x} : X \to [0,1]$ ($\mu_{L^u} : U \to [0,1]$) together with a maximum point at $c(L)$ called *core position*, is associated, which also have to fulfill the following conditions:

- (*Convexity of support*)
  $\mu_L^x(x)$ monotonically increasing for $x < c_x(L^x)$
  $\mu_L^x(x)$ monotonically decreasing for $x > c_x(L^x)$
- (*Partitioning*)
  For $L^x = \{L_1^x, \ldots, L_k^x\}$ and every $x \in X$, it holds that $\sum_{j=1}^{k} \mu_{L_j^x}(x) > 0$
- (*Feedback Correspondence*)
  $\mu_{L_i^x}(c_x(L_i^x)) > 0$ and $\mu_{L_i^x}(c_x(L_j^x)) = 0$ for $j \neq i$.

While they are written down for state variables only, to save space, the first two conditions have to hold also for input variables. The requirement of feedback correspondence is made only for time-discrete systems and ensures that a RFS behaves like a *finite automaton* on its core position vectors, where the transition relation of the automaton is given by the rule base of the system.

Usually, in *standardized RFS*, triangular and ramp-shaped membership functions are employed, where the core positions of ramp-shaped membership functions are at the intersection point of the constant part with the triangular part of the membership function. Thus, the following condition holds for standardized RFS:

$$\forall x \in X, \sum_{L_i^x \in L^x} \mu_{L_i^x}(x) = 1. \tag{7}$$

In the case of MIMO-systems, the above granulation of the input/state-space is carried out component-wise.

The rules have the form:

$$\begin{aligned} \text{IF } u_{r_1} = L_{k_1}^{u_{r_1}} \text{ AND } \ldots \text{ AND } u_{r_j} = L_{k_j}^{u_{r_j}} \\ \text{AND } x_{r_1} = L_{k_1}^{x_{r_1}} \text{ AND } \ldots \text{ AND } x_{r_i} = L_{k_i}^{x_{r_i}} \\ \text{THEN } x_{p_1} = L_{k_1}^{x_{p_1}} \text{ AND } \ldots \text{ AND } x_{p_l} = L_{k_i}^{x_{p_l}}. \end{aligned} \tag{8}$$

For the rest of this paper, we will omit the explicit statement of the variables within the rules, as the linguistic values themselves already note to which variable they belong.

The linguistic values and core position values can each be combined to vectors q, p with m and n components, for the input and state space respectively, so that rules can be written in a more compact form:

$$R_r : \text{ IF } L_{I_q}^u \text{ AND } L_{I_p}^x \text{ THEN } L_{I_p}^x. \tag{9}$$

Using this form, a rule can also be interpreted as a mapping

$$w : \prod_{i=1}^{m} I_i \times \prod_{j=1}^{n} I_j \to \prod_{k=1}^{n} I_k, \tag{10}$$

which maps the index vectors representing the linguistic value for each input and state variable in the premise of a rule onto the index vectors that represent the recommended values in each conclusion.

After fuzzification of the input values x_i and u_j yields membership functions

$$\begin{aligned}&\mu^{x_i}_{L^{x_i}_1}(x_i),\ldots,\mu^{x_i}_{L^{x_i}_{k_i}}(x_i)\\&\mu^{u_j}_{L^{u_j}_1}(u_j),\ldots,\mu^{u_j}_{L^{u_j}_{k_j}}(u_j),\end{aligned}\tag{11}$$

the activation of each rule, that is the extent to which the input and state values fulfill the respective rule premise, is calculated in the aggregation step to

$$\mu_{q,p}(x,u)=\prod_i \mu^{x_i}_{q_i}(x_i)\prod_j \mu^{u_j}_{p_j}(u_j).\tag{12}$$

In the implication step of the inference procedure, the membership function of every possible output value y, i.e. its singleton membership function, is weighted with the activations of the rules that contain the output value in their conclusion:

$$\mu_{w(q,p)}(x,u,y)=\mu_{q,p}(x,u)\cdot\mu^{\text{Singleton}}_{w(q,p)}(y).\tag{13}$$

Thereafter, the contribution of each rule R_k to the activation of every possible output value y is accumulated:

$$\begin{aligned}\mu^{\text{acc}}(x,u,y)&=\sum_{w(q,p)}\mu_{w(q,p)}(x,u,y)\\&=\sum_{w(q,p)}\mu_{q,p}(x,u)\cdot\mu^{\text{Singleton}}_{w(q,p)}(y).\end{aligned}\tag{14}$$

Finally, a crisp output value is produced from the CoS-defuzzification method. Summarizingly, this yields:

$$x(k+1)=\frac{\sum\limits_{q,p}c_x(L^x_{w(q,p)})\cdot\mu_{w(q,p)}(x,u)}{\sum\limits_{q,p}\mu_{w(q,p)}(x,u)}.\tag{15}$$

It was demonstrated in Kempf [2004], that if condition (7) holds, (15) can be simplified to:

$$x(k+1)=\sum_{q,p}c_x(L^x_{w(q,p)})\prod_i\mu^{x_i}_{q_i}(x_i)\prod_j\mu^{u_j}_{p_j}(u_j).\tag{16}$$

This is the standard computational framework for RFS which is based on classical fuzzy inference. The remainder of this paper is devoted to the extension of this framework to two-string RFS that process classical, as well as negative fuzzy rules.

3. RECURRENT TWO-STRING FUZZY SYSTEMS

3.1 Singleton-Based Hyperinference and -defuzzification

For simplicity we assume at first that the configuration of membership functions in both inference strings are equal (this assumption will be relaxed in the end of this section). First, the inference result of both strings have to be combined according to a Hyperinference strategy. While strong and weak veto strategy lead to an output calculated only by positive activations, which are nullified when negative activation exceeds a certain level (zero for strong and positive activation value for weak veto), fuzzy veto is calculated by combining activation and inhibition tendencies with a fuzzy AND-operator, for example algebraic product, minimum or bounded difference.

In this work, we consider strong, weak and fuzzy-difference veto. The support of the admissible output membership function that results from a given veto strategy can be characterized by:

$$x\in\text{supp}(\mu(X))\Leftrightarrow\begin{cases}\mu^-(x)=0 & \text{strong}\\ \mu^+(x)-\mu^-(x)\geq 0 & \text{weak}\\ \mu^+(x)\wedge\neg\mu^-(x)\geq 0 & \text{fuzzy.}\end{cases}\tag{17}$$

Only output values which are contained in this set are admissible output values according to the Hyperinference strategy. That implies that the activation value of singletons, which are contained in the support of this function, can simply be scaled according to the value of the negative output membership function value at the singleton position and the chosen Hyperinference strategy.

However, output singletons not contained in this set have to be deactivated, and additionally it is to be ensured that the defuzzified output value must be contained in the support of the combined output membership function. The idea to solve this problem and still be able to use the convenient CoS-defuzzification method is to calculate an output value for each admissible region, i.e. connected subset of the support of the output membership function, and then apply a choice function, selecting, for example, the output with maximum output membership value.

More precise, if we let the range of admissible outputs be denoted $X=[x^{\min},x^{\max}]$ and the associated linguistic values be denoted $L^x(X)=\{L^x_1,\ldots,L^x_p\}$, we can write for the support of μ on X:

$$\begin{aligned}I&=\text{supp}(\mu(X))\\&=\{I_1,\ldots,I_j\}\end{aligned}\tag{18}$$

where I_k, I_l are mutually disjoint subintervals of X for $k\neq l$.

First, the partitioning $I=\{I_1,\ldots,I_j\}$ of $\text{supp}(\mu(X))$ has to be calculated. In the case of strong veto Hyperinference, $\text{supp}(\mu(X))$ consists of compact intervals with endpoints contained in $c_x(L^x(X))$, i.e. core positions, and can be calculated for each variable separately using Algorithm 1.

Algorithm 1 Calculating I for strong veto Hyperinference

```
j = 1, i = 1
while i < |c_x(L^x(X))| do
    if μ⁻(c_x(L^x_i)) == 0 then
        put c_x(L^x_i) in I^L_j
    else
        if I^L_j ! = ∅ then
            j = j + 1
        end if
    end if
    i = i + 1
end while
for l ∈ {1, ..., j} do
    set I_l^{x^min} = (I^L_l)_1
    set I_l^{x^max} = (I^L_l)_{|I^L_l|}
    set I_l = [I_l^{x^min}, I_l^{x^max}]
end for
```

However, when employing weak or fuzzy veto Hyperinference, the endpoints of a subinterval I_k might not be core positions anymore. In the case of weak veto and difference veto, these endpoints are given by the intersection points of $\mu^+(X)$ and $\mu^-(X)$. Thus, we have to create virtual singletons c^* at these boundary points. The activation of these singletons, i.e. their output membership values, can be calculated by interpolating the positive output membership function linearly between neighboring singleton positions. This has the advantage that some local information about the recommendations in the forbidden region is used for

calculation of the output. In this situation, Algorithm 2 is employed to calculate the partitioning of the admissible region in the output space, again separately for each state variable. The virtual corepositions are calculated as the in-

Algorithm 2 Calculating I for weak and fuzzy veto Hyperinference

```
j=1,i=1
while i < |c_x(L^x(X))| do
    if μ(c_x(L_i^x)) == 0 then
        if I_j^L != ∅ then
            calculate c_{I_j^L}^right
            put c_{I_j^L}^right in I_j^L
            j = j + 1
        end if
    else
        if I_j^L == ∅ then
            calculate c_{I_j^L}^left
            put c_{I_j^L}^left in I_j^L
        end if
        put c_x(L_i^x) in I_j^L
    end if
    i = i + 1
end while
for l ∈ {1, ..., j} do
    set I_l^{x^min} = (I_l^L)_1
    set I_l^{x^max} = (I_l^L)_{|I_l^L|}
    set I_l = [I_l^{x^min}, I_l^{x^max}]
end for
```

tersection point between $\mu^+(X)$ and $\mu^-(X)$ on a compact interval bounded by two corepositions c_1, c_2 according to

$$x_c^i = \frac{c_1 \cdot \delta\mu(c_2) - c_2 \cdot \delta\mu(c_1)}{\delta\mu(c_2) - \delta\mu(c_1)}, \tag{19}$$

where

$$\delta\mu(x) = \mu^+(x) - \mu^-(x). \tag{20}$$

The output membership value at this point is then calculated by interpolating $\mu^+(X)$ between neighboring corepositions:

$$\mu^+(x_c^i) = \frac{c_1 - x_c^i}{c_1 - c_2} \cdot \mu^+(c_1) + (1 - \frac{c_1 - x_c^i}{c_1 - c_2}) \cdot \mu^+(c_2). \tag{21}$$

Then, the system output can be calculated as:

$$x^{\text{res}} = \Psi(x_{I_1}^{\text{res}}, \ldots, x_{I_j}^{\text{res}}), \tag{22}$$

with some choice function Ψ, and

$$x_{I_k}^{\text{res}} = \frac{\sum\limits_{c_p^x \in L_{I_k}^x} c_p^x \cdot \mu_{\text{agg}}^x(c_p^x)}{\sum\limits_{c_p^x \in L_{I_k}^x} \mu_{\text{agg}}^x(c_p^x)}. \tag{23}$$

Reasonable choices for Ψ would be selecting the subinterval that contains the largest membership function value or the largest area under the membership function.

Now it is easy to see why assuming equal output membership function configurations in both inference strings was unnecessary. It suffices to assume that, for each state variable, the largest and the smallest coreposition of the negative inference string are enclosed by corepositions of the positive string. Since it is possible to interpolate the output membership functions linearly between neighboring core positions, every value that is a coreposition in one inference string but not the other can, in the other inference string, be represented by a virtual coreposition during Hyperinference. Input membership functions, on the other hand, can be selected completely independent from another in each inference string to achieve even greater flexibility in the implementation of warnings.

4. APPLICATION

4.1 Simulations

We consider a biologically-inspired system, namely the population dynamics of a given species. This example was also used for illustrative purposes in Kempf [2004] and Adamy and Kempf [2003]. Therein, the dynamic change of the size of an insect population in every year is modelled with respect to the amount of nutrition available to members of this particular species, as well as the population size, in the preceding year. This results in a model with non-overlapping generations. By employing a recurrent fuzzy system for modelling, it is possible to convert known ecological effects, e.g. intraspecific concurrence, into rules which describe the behavior of the system in different regions of the state space.

In Kempf [2004], this model serves as a recurring, illustrative example, which is instanciated with different parameters or rule combinations to illustrate different effects, for example chaos, which can occur in recurrent fuzzy systems. However, the basic structure of the model is as following: The size of the population x as well as the amount of nutrition u, which is available each year, are modelled on a logarithmic scale reaching from 10^4 to 10^8, implying that the corresponding variables take values in the interval $[4, 8]$. For each variable, three different linguistic values are used: $L^x = L^u = \{\text{small}, \text{medium}, \text{large}\}$, which are associated to core positions $c_x(L^x) = c_u(L^u) = \{4, 6, 8\}$. A complete rulebase for this system thus consists of 9 rules. We use a triangular membership function for 'medium', and ramp-shaped membership function for 'small' and 'large', respectively.
Moreover, the following rulebase is employed:

$$\begin{aligned}
&R_1^+ : \text{IF } x \text{ sm AND } u \text{ sm THEN } x \text{ sm}\\
&R_2^+ : \text{IF } x \text{ sm AND } u \text{ med THEN } x \text{ sm}\\
&R_3^+ : \text{IF } x \text{ sm AND } u \text{ lar THEN } x \text{ med}\\
&R_4^+ : \text{IF } x \text{ med AND } u \text{ sm THEN } x \text{ med}\\
&R_5^+ : \text{IF } x \text{ med AND } u \text{ med THEN } x \text{ lar}\\
&R_6^+ : \text{IF } x \text{ med AND } u \text{ lar THEN } x \text{ sm}\\
&R_7^+ : \text{IF } x \text{ lar AND } u \text{ sm THEN } x \text{ med}\\
&R_8^+ : \text{IF } x \text{ lar AND } u \text{ med THEN } x \text{ lar}\\
&R_9^+ : \text{IF } x \text{ lar AND } u \text{ lar THEN } x \text{ med.}
\end{aligned} \tag{24}$$

For the simulations, the input, i.e. amount of food, is kept at a constant level and the system dynamics is simulated for 25 cycles.

Suppose the described system overall resembles the sought dynamics well, except for locally constrained effects that are known to have been missed, e.g. not covered by

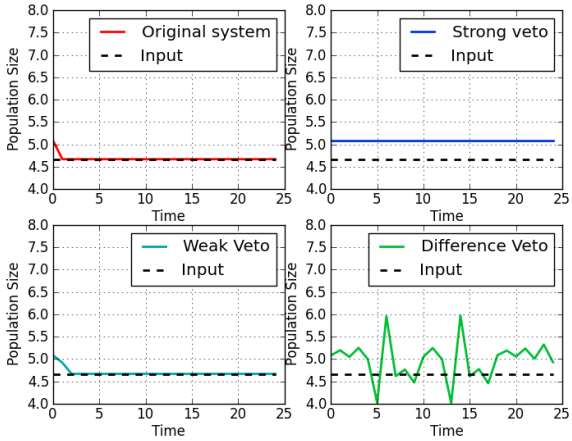

Fig. 2. Different system responses to a small amount of food with small initial population size.

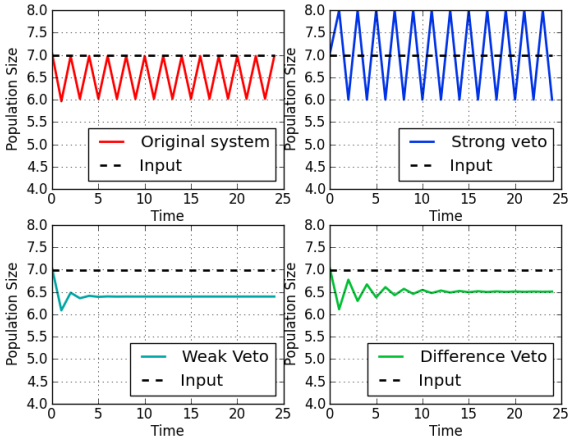

Fig. 3. Different system responses to a high amount of food with large initial population size.

available training data or influences that have not been modelled explicitely.

When the original rule base is altered, the overall behavior of the system will change, which may be unfavorable. In this case, the system can be fine tuned through negative rules. This is illustrated in the following two examples.

In the first example, we are concerned with the behavior of the system for rather low input and initial values. Amount of food is moderately higher than low and initial population size only slightly higher than low. This causes the system dynamics to be attracted towards an equilibrium where amount of available nutrition equals population size. Suppose we know the original system to behave differently in this region of the state space, for example there is an additional source of nutrition that is available only to small populations causes population size to increase for small initial values, regardless of the modelled amount of available food. On the other hand, when the population size exceeds a certain level, intraspecific concurrence will cause the freshly increased population to decrease again. To this end, the following negative rules are introduced in a separate, negative inference string which is parametrized in the same way as the original system:

$$\begin{aligned} R_1^- &: \text{IF } x \text{ sm AND } u \text{ sm THEN } x \text{ sm} \\ R_2^- &: \text{IF } x \text{ med AND } u \text{ sm THEN } x \text{ med} \end{aligned} \tag{25}$$

Note, that both negative rules directly contradict their positive counterparts. The resulting response dynamics for $u_0 = 4.669, x_0 = 5.082$ of the original system, as well as the different veto strategies are depicted in Figure 1. While weak veto changes the systems response only slightly, in that it slows convergence to the equilibrium down, system behavior is changed qualitatively for strong and fuzzy veto. Strong veto actually behaves pathological, as a forbidden output value is produced. However, this is due to the contradicting rules which cause the combined membership function to be identically zero on the whole output space. Since this problem is fundamentally unsolvable without an additional conflict resolution mechanism, the defuzzification procedure was designed such that in this case, the output is kept constant. While we have included this case for the sake of completeness, it is unlikely that such a rule base combination would be employed in practice. However, fuzzy veto is able to produce a reasonable output, but this comes at the cost of possibly chaotic behavior.

In the second example, we milden the effects of intraspecific concurrence for rather high population sizes, for example because of decreased population or food density. Therefore, we modify the system dynamics for relatively high nutrition amounts and population sizes by means of the following negative rules:

$$\begin{aligned} R_1^- &: \text{IF } x \text{ lar AND } u \text{ med THEN } x \text{ sm} \\ R_2^- &: \text{IF } x \text{ lar AND } u \text{ lar THEN } x \text{ sm} \\ R_3^- &: \text{IF } x \text{ med AND } u \text{ lar THEN } x \text{ med.} \end{aligned} \tag{26}$$

The different response dynamics of the system were simulated for the initial values $u_0 = 7.029, x_0 = 6.993$. Simulation results are presented in Figure 2. In this situation, the original model harmonically oscillates in $[6, 7]$. Processing the proposed negative rules with strong veto increases the area that is spanned by the oscillation to $[6, 8]$. This is due to the effect of R_3^-, which forbids medium population sizes when the amount of available food is high and the population is already medium-sized. Weak and fuzzy veto show distinctly different behavior. Their effect of the system dynamics apparently establishes a stable equilibrium around $x = 6.5$, to which both systems converge at slightly different rates.

5. CONCLUSION

5.1 Discussion

We found that unrestricted combinations of recommendations and warnings can cause pathological effects. Contradicting rules, especially for strong veto, can cause the output membership function to vanish identically on the whole range of output variables. In this case, it is not clear what the system is supposed to do, so if one wants to keep the possibility of designing contradicting rulebases, there is a need to introduce conflict resolution strategies which, if hard-wired into the system, may in turn also cause pathological behavior.

However, simulations have shown that our singleton-based approach to Hyperinference offers a flexible and convenient way of realizing two-string inference in recurrent fuzzy systems (or, more generally, fuzzy inference systems working with CoS-defuzzification), rendering two-string RFS feasible in practical applications, on which we give a brief overview in the remainder of this section.

Control Engineers are often interested in enforcing additional constraints upon their controllers, e.g. cancelling steady-state-error or realizing anti-windup mechanisms. Two-string inference makes it possible to realize such mechanisms conveniently without altering the existing system (see Kiendl [1997], or Krause [1999]).

When fuzzy inference systems are applied to pattern recognition tasks, pattern primitives are encoded as fuzzy sets in the input space, which then can be combined to more complex patterns. RFS are successfully applied to sequential pattern recognition tasks, where a specific, temporally ordered sequence of patterns is to be recognized. For the additional complexity of considering patterns appearing in specific order, generally, robust systems are necessary. Guarracino et al. [2013] have employed two-string fuzzy systems to encode false positives, i.e. prominent 'non-examples' of specific patterns, in a separate rule base. This increases system performance while retaining the interpretable representation of the original pattern, since the positive rule base is not altered. This suggests exploiting the same idea in RFS-based sequential pattern recognition systems.

5.2 Future Work

We based our method on techniques described in Kiendl [1997]. Another approach to inference with positive as well as negative rules are the p/n fuzzy systems, introduced in Branson and Lilly [2001]. These methods have not been directly compared yet. While both methods share some similarities, they are based on slightly different requirements to negative inference, which may result in different results, especially in some special cases.

Furthermore, since we demonstrated the different Hyperinference strategies to affect the systems behavior in distinctly different ways, it would be convenient to have means to combine the positive and negative activations to a combined output membership function in a more flexible way than it is possible with Hyperinference veto strategies.

Since we have shown two-string inference to be useful for modelling complex systems, another possibility lies in automated design of complete two-string systems, e.g. by genetic optimization methods. Since humans are accustomed to rule systems which also encompass prohibitions, e.g. "If traffic light is red, do not cross the street", such system models would be capable of modelling highly nonlinear local effects with a fewer number of rules than a conventional system would have required.

REFERENCES

J. Adamy. Breakout prediction for continuous casting by fuzzy mealy automata. In *Proc. 3rd European Congress of Intelligent Techniques and Soft Computing (EUFIT '95), Aachen, Germany*, 1995.

J. Adamy and R. Kempf. Regularity and chaos in recurrent fuzzy systems. *Fuzzy Sets and Systems*, 140, 2003.

J.S. Branson and J.H. Lilly. Incorporation, characterization, and conversion of negative rules into fuzzy inference systems. *IEEE Transactions on Fuzzy Systems*, 9 (2), 2001.

A. Flemming. *Zeitkontinuierliche rekurrente Fuzzy-Systeme*. Düsseldorf: VDI Verlag, 2008.

V. Gorrini and H. Bersini. Recurrent fuzzy systems. In *Proceedings of the Third IEEE Conference on Fuzzy Systems, Orlando, Florida*, 1994.

M.R. Guarracino, R. Jasinevicius, R. Krusinskiene, and V. Petrauskas. Fuzzy hyperinference-based pattern recognition. In C. Borgelt, M.Ã. Gil, J.M.C. Sousa, and M. Verleysen, editors, *Towards Advanced Data Analysis by Combining Soft Computing and Statistics*. Berlin; Heidelberg: Springer, 2013.

R. Kempf. *Rekurrente Fuzzy-Systeme*. Düsseldorf: VDI Verlag, 2004.

R. Kempf and J. Adamy. Equilibria of recurrent fuzzy systems. *Fuzzy Sets and Systems*, 140:231–257, 2003.

R. Kempf and J. Adamy. Sequential pattern recognition employing recurrent fuzzy systems. *Fuzzy Sets and Systems*, 146, 2004.

H. Kiendl. *Fuzzy Control methodenorientiert*. München; Wien: Oldenbourg Verlag, 1997.

H. Kiendl. Advanced fuzzy systems and applications. In W. Bauer, editor, *Fuzzy-Neuro-Systeme '98, computational intelligence : proceedings of the 5. International Workshop Fuzzy-NeuroSystems '98 (FNS '98), Munich, Germany*, 1998.

E. P. Klement, L. T. Koczy, and B. Moser. Are fuzzy systems universal approximators? *International Journal of General Systems*, 28(2-3), 1999.

P. Krause. Demonstration of the use of hyperinference: Position control using an electric disc-motor. In B. Reusch, editor, *Computational Intelligence - Theory and Applications International Conference, 6th Fuzzy Days Dortmund, Germany*. Springer Berlin Heidelberg, 1999.

J.H. Lilly. Evolution of a negative-rule fuzzy obstacle avoidance controller for an autonomous vehicle. *IEEE Transactions on Fuzzy Systems*, 15(4), 2007.

T. M. Ngyuen and Q. M. J. Wu. A combination of positive and negative fuzzy rules for image classification problem. In *Proceedings of the Seventh International Conference on Machine Learning and Applications (ICMLA '08), San Diego, California*, 2008.

U. Schwane, J. Praczyk, and H. Kiendl. Adaption von Reglerparametern unter Verwendung von datenbasiert generierten positiven und negativen Fuzzy-Regeln mit Konklusionen mit mehreren Ausgangsgrössen. In H. Kiendl and Ch. Frenck, editors, *7. GMA-Workshop "Fuzzy-Control", Forschungsberichte der Fakultät für Elektrotechnik, Nr. 0397, Universität Dortmund*, 1997.

A. Schwung. *Modellbildung und Fehlerdiagnose mit rekurrenten Fuzzy-Systemen*. Düsseldorf: VDI Verlag, 2011.

H. Surmann and M. Maniadakis. Learning feed-forward and recurrent fuzzy systems: a genetic approach. *Journal of Systems Architecture*, 47, 2001.

H. Surmann and L. Peters. Moria - a robot with fuzzy controlled behaviour. In D. Driankov and A. Saffiotti, editors, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*. Physica-Verlag, 2000.