Designing a Self-adaptive Union-Based Rule-Antecedent Fuzzy Controller Based on Two Step Optimization

Chang-Wook Han and Jung-Il Park

School of Electrical Engineering and Computer Science, Yeungnam University,
214-1 Dae-dong, Gyongsan, Gyongbuk, 712-749 South Korea
{cwhan, jipark}@yumail.ac.kr

Abstract. A self-adaptive union-based rule-antecedent fuzzy controller (SURFCon), which can guarantee a parsimonious knowledge base with reduced number of rules, is proposed. The SURFCon allows union operation of input fuzzy sets in the antecedents to cover bigger input domain compared with the complete structure rule which consists of AND combination of all input variables in its premise. To construct the SURFCon, we consider the union-based logic processor (ULP) which consists of OR and AND fuzzy neurons. The fuzzy neurons exhibit learning abilities as they come with a collection of adjustable connection weights. In the development stage, genetic algorithm (GA) constructs a Boolean skeleton of SURFCon, while stochastic reinforcement learning refines the binary connections of GA-optimized SURFCon for further improvement of the performance index. A cart-pole system is considered to verify the effectiveness of the proposed method.

1 Introduction

Fuzzy logic, proposed by Zadeh in 1965, is a logic with fuzzy truth, fuzzy connectives and fuzzy rules of inference rather than the conventional two-valued or even multi-valued logic [1]. It combines multi-valued logic, probability theory and a knowledge base to mimic human thinking by incorporating the uncertainty inherent in all physical systems. Relying on the human nature of fuzzy logic, an increasing number of successful applications have been developed, like automatic process control [2], pattern recognition systems [3] and so forth.

A common practice in traditional approaches to building fuzzy rule bases is to use all AND-combinations of input fuzzy sets as rule antecedents [4][5]. In this case, the number of such combinations increases exponentially with the input number [6]. To reduce the size of the rule bases, a self-adaptive union-based rule-antecedent fuzzy controller (SURFCon) is proposed in this paper. The SURFCon allows union operation of input fuzzy sets in the antecedents to cover bigger input domain compared with the complete structure which consists of AND combinations of fuzzy sets of all input variables in its premise [7]. Basically, the SURFCon is constructed with the aid of union-based logic processor (ULP) which consists of OR and AND fuzzy neurons. The fuzzy neurons exhibit learning abilities as they come with a collection of adjustable connection weights [6][8]. This paper aims at constructing a binary structure of
Designing a SURFCon Based on Two Step Optimization

SURFCon by genetic algorithm (GA) [9], and subsequently further refining the binary connections by random signal-based learning employing simulated annealing (SARSL) introduced in [10][11]. The effectiveness of the SURFCon is demonstrated by stabilizing an inverted pendulum system.

2 Structure of the SURFCon

Before discussing the architecture of the SURFCon, we will briefly remind AND and OR fuzzy neurons and then move on to the ULP which is the basic logic processing unit of the SURFCon.

2.1 OR and AND Fuzzy Neurons

As originally introduced in [6][8], fuzzy neurons emerge as result of a vivid synergy between fuzzy set constructs and neural networks. In essence, these neurons are functional units that retain logic aspects of processing and learning capabilities characteristic for artificial neurons and neural networks. Two generic types of fuzzy neurons are considered:

- **AND neuron** is a nonlinear logic processing element with n-inputs \( x \in [0, 1]^n \) producing an output \( y \) governed by the expression

  \[
  y = \text{AND}(x; w)
  \]

  where \( w \) denotes an n-dimensional vector of adjustable connections (weights). The composition of \( x \) and \( w \) is realized by an t-s composition operator based on t- and s-norms, that is

  \[
  y = \prod_{i=1}^{n} (w_i, s, x_i)
  \]

  with “s” denoting some s-norm and “t” standing for a t-norm. As t-norms (s-norms) carry a transparent logic interpretation, we can look at as a two-phase aggregation process: first individual inputs (coordinates of \( x \)) are combined or-wise with the corresponding weights and these results produced at the level of the individual aggregation are aggregated and-wise with the aid of the t-norm.

  By reverting the order of the t- and s-norms in the aggregation of the inputs, we end up with a category of OR neurons,

  \[
  y = \text{OR}(x; w)
  \]

  that is

  \[
  y = \bigvee_{i=1}^{n} (w_i, t, x_i)
  \]

  We note that this neuron carries out some and-wise aggregation of the inputs followed by the global or-wise combination of these partial results.

  Some obvious observations hold:
(i) For binary inputs and connections, the neurons transform to standard OR and AND gates.
(ii) The higher the values of the connections in the OR neuron, the more essential the corresponding inputs. This observation helps eliminate irrelevant inputs; the inputs associated with the connections whose values are below a certain threshold are eliminated. An opposite relationship holds for the AND neuron; here the connections close to zero identify the relevant inputs.
(iii) The change in the values of the connections of the neuron is essential to the development of the learning capabilities of a network formed by such neurons; this parametric flexibility is an important feature to be exploited in the design of the networks.

These two types of fuzzy neurons are fundamental building blocks used in the design of logic expressions supporting the development of logic-driven models.

2.2 SURFCon and Its Two-Step Optimizations

To construct the SURFCon, we first elaborate on the ULP which consists of OR and AND fuzzy neurons, as shown in Fig. 1, where $\mu_N(x_i)$, $\mu_Z(x_i)$ and $\mu_P(x_i)$ are the membership grades of the fuzzy sets N (negative), Z (zero) and P (positive) for the input variable $x_i$, $i=1,2,3,4$, respectively. The OR and AND fuzzy neurons realize pure logic operations on the membership values and exhibit learning abilities as being introduced in [6][8]. In this paper, we consider these triangular norms and co-norms to be a product operation and probabilistic sum, respectively.

![Fig. 1. Structure of an ULP](image)

An important characteristic of ULP is that union operation of input fuzzy sets is allowed to appear in their antecedents, i.e., incomplete structure. For fuzzy system of complex processes with high input dimension, the ULP is preferable because it achieves bigger coverage of input domain compared with the complete structure. For example, consider a system with $x_1$, $x_2$ as its inputs and $y$ as its output characterized by three linguistic terms, N, Z and P, respectively. The incomplete structure rule ‘If $x_1=N$ then $y=N$’ covers the following three complete structure rules:
(i) If \((x_1=N)\) and \((x_2=N)\) then \(y=N\)
(ii) ‘If \((x_1=N)\) and \((x_2=Z)\) then \(y=N\)
(iii) ‘If \((x_1=N)\) and \((x_2=P)\) then \(y=N\)

Similarly, the rule ‘If \((x_1=N\) or \(Z)\) and \((x_2=N\) or \(Z)\) then \(y=N\)’ covers the following four complete structure rules:

(i) If \((x_1=N)\) and \((x_2=N)\) then \(y=N\)
(ii) If \((x_1=N)\) and \((x_2=Z)\) then \(y=N\)
(iii) If \((x_1=Z)\) and \((x_2=N)\) then \(y=N\)
(iv) If \((x_1=Z)\) and \((x_2=Z)\) then \(y=N\)

Fig. 2. Structure of a SURFCon with 4 input and 1 output variables characterized by 3 and 5 fuzzy sets, respectively (NU=20)

Fig. 2 describes the SURFCon constructed with the aid of ULPs. The OR neurons in the output layer are placed to aggregate the outputs of ULPs for each corresponding consequences. In Fig. 2, the connections to the ULPs are described as bold lines which contain a set of connection lines, refer to Fig. 1. The only parameter that has to be controlled in the SURFCon is the number of ULP (NU), which will be set large enough in the experiment. A conflict occurs in the rule bases if there exist two rules which have overlapping AND combinations but different linguistic consequences. Let us consider the following two rules to count the number of conflict (NC):

(i) If \((x_1=N\) or \(Z)\) and \((x_2=N\) or \(Z)\) then \(y=N\)
(ii) If \((x_1=Z\) or \(P)\) and \((x_2=N\) or \(Z)\) then \(y=Z\)

In this case, NC=2 because the antecedents ‘\((x_1=Z)\) and \((x_2=N)\)’ and ‘\((x_1=Z)\) and \((x_2=Z)\)’ have different consequences. Therefore, NC should be checked for all possible pairs of different consequences of the rule bases. It will be included in evaluating the performance index to remove the conflict from the rule bases later on.

For the development of the SURFCon, GA attempts to construct a Boolean structure of SURFCon by selecting the most essential binary connections that shape up the architecture. GA optimizes the binary connections \(W\) and \(u\), as shown in Fig. 2. All the connections to AND neurons, \(h_k\), in the ULPs are initialized as zero (valid connection). It is worth noting that if the connection weights from \(F_i\) to an OR neuron in the ULP are all-one, that is, \(x_i\) is ‘don’t care’ in this ULP, the corresponding connection to AND neuron should be modified as one (invalid connection). It is obvious that the following two cases lead to an invalid rule antecedent:
(i) All-one connection weights to an ULP, meaning that all input variables are ignored in the antecedent.

(ii) All-zero connection weights to an OR neuron in the ULP, i.e., empty fuzzy set for the corresponding input variable.

Therefore, these ULPS will be removed from the rule bases and finally the compact rule bases will be established. To avoid the rule confliction, the output of an ULP should be connected to only one of the OR neurons in the output layer. Since the GA is a very popular optimization algorithm considered in many application areas, we do not elaborate on this. For more details about the GA, please refer to [9].

Once a Boolean skeleton has been constructed by GA, we concentrate on the detailed optimization of the valid binary connections with the aid of SARSL. It is apparent that the number of valid binary connections is much smaller than the number of all possible connections in the SURFCon. Therefore, the SARSL that has an excellent local search ability proved in [10][11] is considered to refine the reduced number of valid binary connections optimized by GA. The SARSL refinement involves transforming binary connections into the weights in the unit interval. This enhancement aims at further improvement in the value of the performance index. Obviously we do not claim that the SARSL is the most effective learning method for this purpose. We intend to show how much the connection refinement affects the performance of the SURFCon. For more details about SARSL, please refer to [10][11].

3 Experimental Results

To show the effectiveness of the proposed SURFCon, a cart-pole system, a well-known nonlinear system, was considered. The objective is to bring the cart to center with a vertical pole. The system has four state variables which are \( \theta \) (angle of the pole with the vertical), \( \dot{\theta} \) (angular velocity of the pole), \( x \) (position of the cart) and \( \dot{x} \) (linear velocity of the cart). The nonlinear differential equations for a cart-pole system are described as follows [7][12]:

\[
\ddot{\theta} = \frac{(M + m)g \sin \theta - \cos \theta [f + mL \dot{\theta}^2 \sin \theta - \mu_s \text{sgn}(\dot{x})]}{\frac{4}{3}(M + m)L - mL \cos^2 \theta} - \frac{\mu_s (M + m) \dot{\theta}}{mL}
\]

\[
\dot{x} = f + mL(\dot{\theta}^2 \sin \theta - \dot{\theta} \cos \theta) - \mu_s \text{sgn}(\dot{x})
\]

The parameter values used in this simulation were set as the same as [11] except the failure conditions, where \(|\theta| > 0.2\text{rad}\) or \(|x| > 0.5\text{m}\). The following optimization parameters were considered: (i) GA parameters are population size=100; generation number=200; crossover rate=0.9; mutation rate=0.03; (ii) SARSL parameters are learning rate=0.01; initial temperature=0.1; cooling rate=0.99; iteration number=500; (iii) others are time step=0.01s; simulating number of time steps for each initial condition q=500; NU=20. For the GA, standard version including two-point crossover based on the boundary between binary (connections \( W \) for antecedents) and integer
Designing a SURFCon Based on Two Step Optimization

745

connections \( u \) for consequences) codes was used. GA optimized 240 (12x20) binary parameters and 20 integer parameters that have the integer values of \( \{1,2,\ldots,5\} \). The parameters of SARS\( L \) were set to perform fine learning of the binary connections. The following normalized performance index (fitness function) that has to be maximized was used [7]:

\[
Q = \frac{1}{8} \sum_{\text{cond}} \left\{ q_s \left[ 1 - \frac{1}{2q} \sum_{j=1}^8 \left( \frac{\theta_j}{\theta_{\text{fail}}} + \frac{x_j}{x_{\text{fail}}} \right) \right] \right\} \left\{ 1 - \frac{\text{NC}}{\text{NC}_{\text{max}}} \right\}
\]

where \( \theta_{\text{fail}} = 0.2\text{rad}, x_{\text{fail}} = 0.5\text{m}, q_s \) is the survival time steps for each trial, and \( \text{NC}_{\text{max}} \) is the maximum NC which is set as 10. If NC exceeds \( \text{NC}_{\text{max}} \), NC = \( \text{NC}_{\text{max}} \). For the evaluation of the performance index, eight different sets of initial conditions (cond1~cond8) were considered to cover wide range of input spaces. Because our focal point is SURFCon and its two-step optimization, we assume that fuzzy sets of the input/output variables are given in advance as 3/5-uniformly distributed triangular membership functions with an overlap of 0.5 and left unchanged. For the defuzzification, center of area method was used. Ten independent simulations for the optimization of SURFCon have been performed, and the results are shown in Table 1. This table describes the average best performance index after GA and SARS\( L \) over ten independent simulations as well as the maximum, minimum and average number of valid ULP (incomplete structure rules) for ten optimized SURFCons. As can be seen, the optimized SURFCon has at most 15 incomplete structure rules covering most of the essential input domain without linguistic conflict, and besides, the SARS\( L \) further refines the valid binary connections optimized by GA.

Table 1. Results of the optimized SURFCons

<table>
<thead>
<tr>
<th></th>
<th>After GA</th>
<th>After SARS( L )</th>
<th>Max rule</th>
<th>Min rule</th>
<th>Average rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.912</td>
<td>0.947</td>
<td>15</td>
<td>13</td>
<td>14.1</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 3(a) illustrates the results of testing simulations for the optimized SURFCon with a set of initial condition \((\theta, \dot{\theta}, x, \dot{x}) = (0, 0, -0.25, 0)\) which is independent of the sets for the optimization. To compare the results, the conventional fuzzy controller (CFC) which has all possible AND combinations (81 complete structure rules) as rule antecedents was considered. To prove the SURFCon to be practical, we have also performed experiment on the real cart-pole system manufactured by Realgain Co., Ltd. (Fig. 4). The experimental results are shown in Fig. 3(b).

Obviously the performance of CFC is better than that of SURFCon, but the control results indicate that the reduced rule bases of SURFCon are enough to center the cart with a vertical pole, and moreover, the performance of SURFCon outperforms that of [7] and [11]. Both the simulation and experimental results reveal that the SURFCon is really feasible.
Fig. 3. Control results using a testing initial condition: (a) Simulation results, (b) Experimental results

Fig. 4. Real inverted pendulum system

4 Conclusions

The SURFCon that allows union operation to appear in their antecedents for the parsimonious rule bases has been demonstrated. It has been constructed with the aid of fuzzy neurons and two-step optimizations, where GA develops a Boolean skeleton and subsequently SARSL further refines the binary connections. As can be seen from the simulation and experimental results, the incomplete structure of rule allows the SURFCon to utilize only fewer rules without performance degrading, and also proved the SURFCon to be practical.
References