Modeling of hierarchical fuzzy systems

Ming-Ling Lee, Hung-Yuan Chung*, Fang-Ming Yu

Department of Electrical Engineering, National Central University, Chung-Li 32054, Taiwan, ROC

Received 7 September 2000; received in revised form 21 October 2002; accepted 30 October 2002

Abstract

A new kind of mapping rule base scheme is proposed to get the fuzzy rules of hierarchical fuzzy systems. The algorithm of this scheme is developed such that one can easily design the involved fuzzy rules in the middle layers of the hierarchical structure. In contrast with the conventional single layer fuzzy controller, the present method has approximate performance using the same scaling factors. Next, examples are given. At last, simulated results demonstrate that the algorithm is effective and feasible.

© 2002 Elsevier B.V. All rights reserved.

Keywords: Hierarchical fuzzy control; Seesaw system

1. Introduction

The design of fuzzy controllers is commonly a time-consuming activity involving knowledge acquisition, definition of the controller structure, definition of rules, and other controller parameters.

At present, one of the important issues in fuzzy logic systems is how to reduce the total number of involved rules and their corresponding computation requirements. In a standard fuzzy systems, the number of rules increases exponentially with the number of variable increases. Suppose there are \( n \) input variables and \( m \) membership functions for each variable, then it needs \( m^n \) rules to construct a complete fuzzy controller. As \( n \) increases, the rule base will quickly overload the memory and make the fuzzy controller difficult to implement. In fact, the complexity of a problem increases exponentially with the number of variables involved, called “curse of dimensionality”, is not unique to fuzzy systems. Hence, to deal with the “curse of dimensionality” and the rule-explosion problem, the idea of hierarchical fuzzy systems (HFSs) was reported [1,5]. These hierarchical fuzzy systems consist of a number of low-dimensional fuzzy systems in a hierarchical form. The hierarchical fuzzy systems have the advantage that the total number of rules increases only linearly with the number

* Corresponding author. Tel.: +886-3-4227151-4475; fax: +886-3-4255830.
E-mail address: hychung@ee.ncu.edu.tw (H.-Y. Chung).
For the hierarchical fuzzy system in Fig. 1, with \(4(n = 4)\) input variables and \(5(m = 5)\) membership functions, then each low-dimensional fuzzy system consists of \(5^2(m^2)\) rules and, therefore, the total number of rules is \(3 \times 5^2 = 75[(n - 1)m^2]\), which is a linear function of the number input variables \(n\). But, the number of rules in the standard fuzzy system is \(5^4 = 625\). It could be found that the total number of rules is greatly reduced under hierarchical fuzzy systems structure.

Moreover, in hierarchical fuzzy systems (HFSs) \([1–9]\), the intermediate outputs are artificial in nature in many cases and do not possess physical meaning. Thus, for fuzzy logic units (FLUs), if they are used as the input variables of the next layer as the dotted arrow point in Fig. 1, which is the usual case, then the involved fuzzy rules in the middle layers of the hierarchical structure have little physical meaning and consequently are hard to design. This phenomenon becomes prominent as the number of layers grows larger in an HFS.

To overcome the problem that the outputs of intermediate layers do not possess physical meaning and consequently are hard to design, we propose a new kind of mapping rule base schemes to get the HFSs rule bases where the outputs of the previous layer and the inputs of next layers are defined as intermediate mapping variables. Meanwhile, the outputs of the first layer of FLUs and the inputs of the other FLUs are just decided by the intermediate mapping variables. The intermediate mapping variables are the further result described in Section 3. By the intermediate mapping variables, one can easily design the involved fuzzy rules in the middle layers of the hierarchical structure. As such, all of the rule bases of FLUs need not to be redesigned. Also, using this method, we can get the same input–output model as the conventional single layer fuzzy logic system, the total number of the involved rules can be greatly reduced as well as the system stability can be guaranteed.

The rest of the paper is organized as follows. In Section 2, perspective on hierarchical fuzzy control scheme is described. The details of the proposed scheme are given in Section 3. The algorithm is shown in Section 4. The example and simulated results for the seesaw control problem are shown in Section 5. Reducing the number of rules is discussed in Section 6. Finally conclusions are given in Section 7.

2. Perspective on hierarchical fuzzy control scheme

2.1. Conventional single layer fuzzy logic system

One of the important issues in fuzzy logic systems is how to reduce the number of involved rules and their corresponding computation requirements. In fact, the number of fuzzy rules grows
Input variables = 4.
Membership functions = 5.
Rules = $5^4 = 625$.

Fig. 2. Conventional single layer fuzzy logic system.

Input variables = 4.
Membership functions = 5.
Rules = $3 \times 5^2 = 75$.

(a)

(b)

Fig. 3. Hierarchical fuzzy logic system.

exponentially with the number of input variables. Specifically, a single output fuzzy logic system with $n$ input variables and $m$ membership functions defined for each input variable requires $m^n$ number of fuzzy rules. That is, if we have 4 input variables with respective 5 membership functions, then we need $5^4$ number of rules.

2.2. Hierarchical and structure-hierarchical fuzzy logic system

To reduce the number of involved rules may decrease the difficulties of designing, the idea of HFSs has been reported and shown in Fig. 3, where the input variables are put into a collection of low-dimensional fuzzy logic units (FLUs) instead of a single high-dimensional fuzzy logic system. Compared with the standard fuzzy logic system as shown in Figs. 2 and 3, the HFS dramatically reduces the total number of involved fuzzy rules.

In the conventional HFSs, the outputs of FLUs in the previous layer are used as the input linguistic variables of the next layer. The intermediate outputs, however, are artificial in nature in many cases and do not possess physical meaning. Thus if they are used as the input variables of the next layer, then the involved fuzzy rules in the middle of the hierarchical structure have little physical meaning and consequently is hard to design. This phenomenon becomes prominent as the number of layers grows larger in an HFS.
To overcome the problem, a hierarchical fuzzy system referred to as structure-hierarchical fuzzy system (S-HFS) has been reported [2] as shown in Fig. 4. In the first layer of S-HFS, all input variables can be used as the antecedent linguistic variables, since the system variables are usually used as the input variables in this layer. In the second layer, the outputs of the previous layer should not be used as the antecedent linguistic variables because they have little physical meaning. Instead, they are used in the THEN-parts of the fuzzy rules in this layer. The antecedent linguistic variables of the next layer, for example, can be selected from the representatives chosen from the input variables in the previous layer. As a result, only variables with physical meaning are used in the IF-parts of fuzzy rules and hence the resulting fuzzy rules become easy to design. The above description involves the previous HFS issues, Nevertheless, we describe some results in Section 3.

3. Limpid-hierarchical fuzzy system (L-HFS)

In this section, a new method of mapping rule base schemes referred to as limpid-hierarchical fuzzy system (L-HFS) is proposed to get the HFSs rule base. In what follows, an example is given as shown in Fig. 5 and Table 1. The control rule base of the conventional single layer fuzzy logic system is given in Table 1.

From Fig. 5 and Table 1, the three input variables are $\theta$, $\dot{\theta}$, and $x$. $F_1$ and $F_2$ are the first layer fuzzy logic unit (FLU) and the second layer FLU, respectively. In Table 1, the fuzzy linguistic outputs are obtained by random access so as to generate a basis of the mapping rule base. Define each variable with three linguistic terms that $N$ is negative, $Z$ is zero and $P$ is positive. In this way, we can get $27 \times 3 \times 3 = 27$ rules. And the fuzzy rules of Table 1 are as follows:

- **if** $\theta$ is $P$, and $\dot{\theta}$ is $P$, and $x$ is $P$, **then** $u$ is $P$,
- **if** $\theta$ is $P$, and $\dot{\theta}$ is $Z$, and $x$ is $P$, **then** $u$ is $N$,
  
  :  
  
- **if** $\theta$ is $N$, and $\dot{\theta}$ is $N$, and $x$ is $N$, **then** $u$ is $N$.

First, in accordance with Fig. 5, it is shown that the input variables of FLU $F_1$ are $(\theta \ \dot{\theta})$. Then, from Table 1, one fixes $\theta$ and $\dot{\theta}$ to pick out each vertical column and then make the sorting process...
as shown in Fig. 6. Intuitively, there are five kinds of different columns, so, we only require five mapping variables of the intermediate output of variable $u_0$.

Define five mapping variables as A, B, C, D and E, respectively. Table 2 denotes the relationship of these variables.

Next, choose the first column in Table 2 to explain that $\theta$ is P and $\dot{\theta}$ is also P is mapped to the mapping variable A. Obviously, from Table 2, the FLU of $F_1$, shown in Table 3, can be constructed in the if–then rules as follows:
Table 2
The relationship of the involved column of mapping variables

<table>
<thead>
<tr>
<th>x</th>
<th>P</th>
<th>Z</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>P</td>
<td>Z</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Z</td>
<td>Z</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>P</td>
<td>Z</td>
<td>N</td>
</tr>
</tbody>
</table>

Table 3
The rules for $F_1$

<table>
<thead>
<tr>
<th>$\dot{\theta}$</th>
<th>$\theta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>A</td>
</tr>
<tr>
<td>Z</td>
<td>B</td>
</tr>
<tr>
<td>N</td>
<td>C</td>
</tr>
</tbody>
</table>

- if $\theta$ is P, and $\dot{\theta}$ is P, then $u_{\theta}$ is A,
- if $\theta$ is P, and $\dot{\theta}$ is Z, then $u_{\theta}$ is B,
- if $\theta$ is Z, then $u_{\theta}$ is E,

According to Table 2, it is seen that the relationship between $u_{\theta}$ and $x$ is equivalent to the relationship between the mapping variables (A, B, C, D, E) and $x$. So, Table 4 can be built, which is the FLU of $F_2$. And the rules are as follows:

- if $u_{\theta}$ is A, and $x$ is P, then $u$ is P,
- if $u_{\theta}$ is A, and $x$ is Z, then $u$ is Z,
- if $u_{\theta}$ is B, then $u$ is B,
- if $u_{\theta}$ is E, and $x$ is N, then $u$ is N.

The rules of Tables 3 and 4 are what we want for $F_1$ and $F_2$, respectively. From the above discussion, the resultant output is the same as Table 1. In the following section, we list the step of the algorithm for the L-HFS.

Remark 1. For the sake of proving that we can get the same input–output before and after the L-HFS method, given a three-variable input system, by using conventional single layer fuzzy logic system and given a rule base as in Table 1. The output of $u$ can be obtained as follows:

- if $\theta$ is P, and $\dot{\theta}$ is P, and $x$ is P, then $u$ is P.
However, by using L-HFS method, we can get the same output of $u$ from Table 3 and 4, and Fig. 5. The description is as follows:

- if $\theta$ is P, and $\dot{\theta}$ is P, then $u_0$ is A,
- if $u_0$ is A, and $x$ is P, then $u$ is P.

From the above description, it is obvious that we can get the same input–output model before and after the L-HFS decomposition. Moreover, for a three-variable input system, in conventional single layer fuzzy logic system, the total rules are 27 ($3 \times 3 \times 3 = 27$). However, by using L-HFS method, the total rules are 24. Clearly, it is seen from Tables 3 and 4, and Fig. 5. That means there are 9 rules to get the output of $u_0$ and 15 rules to get the output of $u$. As to the six-variable input system, the same input–output model can be obtained before and after the L-HFS decomposition, and the total number of rules will be greatly reduced, which is described in Section 5.

### 4. Algorithm for the L-HFS

As for making the L-HFS, an example of higher dimensional HFSs of the six-variable input system will be executed in the next section, and the procedures of the algorithm are as follows:

1. Determine the form of HFS and label the FLU from $1_i$ to $n_i$ of the $i$th layer as shown in Fig. 7.
2. From Fig. 7, fix the input variables $x_1, x_2$ and use the sorting process of mapping to get the FLU ($1_1$) output mapping variables. Use the same way to execute the FLU output mapping variables of $2_1, 3_1, \ldots, n_1$.
3. From Fig. 7, fix the input variables $x_1, x_2, x_3, x_4$ and use the sorting process to get the FLU ($1_2$) output mapping variables. Use the same way to execute the FLU output mapping variables of $2_2, 3_2, \ldots, n_2$.
4. Repeat the same procedure as in step 3 until the last second layer is reached.
5. In the final layer, we can tabulate the final FLU rules from the involved linguistic outputs of the previous layer’s mapping variables.

### 5. Examples

**Example 1.** A six-variable input of HFS is considered. The structure of HFS is shown in Fig. 8. Randomly choose the control rule base of the single layer fuzzy logic system listed in
Table 5. The objective is to prove that we can get the same input–output before and after the L-HFS method.

We use the algorithm of L-HFS as described in the previous section and, by Tables 6 to 10 to obtain the $F_1$, $F_2$, $F_3$, $F_4$, $F_5$ which are as shown in Fig. 8, respectively.

In Table 5, fix $x_1, x_2$ and use the L-HFS mapping method to get the mapping variable $A$, $B$, $C$, $D$ and construct the rule of $F_1$ as shown in Table 6. The rules are as follows:

- if $x_1$ is $P$, and $x_2$ is $P$, then $u_1$ is $A$,
- if $x_1$ is $P$, and $x_2$ is $Z$, then $u_1$ is $A$,

... 

- if $x_1$ is $N$, and $x_2$ is $N$, then $u_1$ is $D$. 

Table 5
The rules of example

<table>
<thead>
<tr>
<th>x5</th>
<th>x3</th>
<th>x4</th>
<th>x6</th>
<th>x1</th>
<th>x1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6
The rule of F1 with mapping variables A, B, C, D

<table>
<thead>
<tr>
<th>x2</th>
<th>x1</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Z</td>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>N</td>
<td>A</td>
<td>A</td>
<td>D</td>
</tr>
</tbody>
</table>

Table 7
The rule of F2 with mapping variables E, F, G, H

<table>
<thead>
<tr>
<th>x4</th>
<th>x3</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>E</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Z</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>N</td>
<td>G</td>
<td>H</td>
<td>H</td>
</tr>
</tbody>
</table>
In Table 5, by fixing $x_3, x_4$ and using the L-HFS method to get the mapping variables E, F, G, H and construct the rule of $F_2$ as shown in Table 7. The rules are as follows:

- **if** $x_3$ is P, and $x_4$ is P, **then** $u_2$ is E,
- **if** $x_3$ is P, and $x_4$ is Z, **then** $u_2$ is F,
  
  :  
  
- **if** $x_3$ is N, and $x_4$ is N, **then** $u_2$ is H.

Using the same way, the rules of $F_3$ are shown in Table 8 as follows:

- **if** $x_5$ is P, and $x_6$ is P, **then** $u_3$ is I,
- **if** $x_5$ is P, and $x_6$ is Z, **then** $u_3$ is J,
  
  :  
  
- **if** $x_5$ is N, and $x_6$ is N, **then** $u_3$ is I.
For the second layer of $F_4$, through fixing $x_1, x_2, x_3, x_4$ and using this L-HFS mapping method to get the rules of $F_4$ shown in Table 9, the rules are as follows:

- if $u_1$ is A, and $u_2$ is E, then $u_4$ is R,
- if $u_1$ is A, and $u_2$ is F, then $u_4$ is S,
  
  ...  
- if $u_1$ is D, and $u_2$ is H, then $u_4$ is R.

Finally, for the third layer of $F_5$, we can use this proposed decomposition mapping method, to get the final output $u$ shown in Table 10, and the rules are as follows:

- if $u_3$ is I, and $u_4$ is R, then $u$ is P,
- if $u_3$ is I, and $u_4$ is S, then $u$ is P,
  
  ...  
- if $u_3$ is K, and $u_4$ is V, then $u$ is P.

The input variables of each layer of fuzzy logic unit (FLU) use the triangular membership function. Fig. 9 shows the intermediate mapping variables A, B, ..., I, J and R, S, T, U, V. They are the membership functions of $u, x_1, x_2, x_3, x_4, x_5, x_6$ and $u_1, u_2, u_3, u_4$.

From the above L-HFS decomposition mapping description for six-variable input, we can get the same input–output model before and after the proposed method described in Remarks 2 and 3.

**Remark 2.** For six-variable input system, the basis of mapping rules base is to choose the intermediate mapping variables A, B, ..., K and R, S, T, U, V. For example, fix $x_5, x_6$ and use the L-HFS mapping method to get the intermediate mapping variables I, J, K as listed in Table 11, the rules are as follows:

- if $x_5$ is P, and $x_6$ is P, then $u_3$ is I,
- if $x_5$ is P, and $x_6$ is Z, then $u_3$ is J,
- if $x_5$ is P, and $x_6$ is N, then $u_3$ is K,
- if $x_5$ is Z, and $x_6$ is P, then $u_3$ is I,
- if $x_5$ is Z, and $x_6$ is Z, then $u_3$ is I,
- if $x_5$ is Z, and $x_6$ is N, then $u_3$ is I,
- if $x_5$ is N, and $x_6$ is P, then $u_3$ is I,
- if $x_5$ is N, and $x_6$ is Z, then $u_3$ is I,
- if $x_5$ is N, and $x_6$ is N, then $u_3$ is I.

The above rules are just for organizing the Table 8, the rules of $F_3$. Also, using the same way of the L-HFS decomposition mapping method, the other intermediate mapping variables can be obtained.

**Remark 3.** For a six-variable input system, by conventional single layer fuzzy logic system, if we randomly choose a rule in Table 5 as follows:

- if $x_1$ is P, and $x_2$ is P, and $x_3$ is P, and $x_4$ is P, and $x_5$ is P, and $x_6$ is P, then $u$ is P.
By using L-HFS method, one can get the same output $u$ from Tables 6 to 10, and Fig. 8. The description is as follows:

- if $x_1$ is P, and $x_2$ is P, then $u_1$ is A,
- if $x_3$ is P, and $x_4$ is P, then $u_2$ is E,
- if $x_5$ is P, and $x_6$ is P, then $u_3$ is I,
- if $u_1$ is A, and $u_2$ is E, then $u_4$ is R,
- if $u_3$ is I, and $u_4$ is R, then $u$ is P.

From the above description, it is obvious that we can get the same input–output model before and after the L-HFS decomposition. Moreover, in conventional single layer fuzzy logic system of the
Table 11
The intermediate mapping variables of I, J, K for six-variable input system

<table>
<thead>
<tr>
<th>X_5</th>
<th>X_3</th>
<th>X_4</th>
<th>X_6</th>
<th>X_5</th>
<th>X_3</th>
<th>X_4</th>
<th>X_6</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

six-variable input system, the total number of rules is 729 \((3 \times 3 \times 3 \times 3 \times 3 = 729)\). However, by using L-HFS method, the total number of rules is 58, which can be obtained via the above manner and from Tables 6 to 10, and Fig. 8. That means there are 9 rules to get output of \(u_1\), 9 rules to get output of \(u_2\), 9 rules to get output of \(u_3\), 16 rules to get output of \(u_4\), and 15 rules to get output of \(u_5\), it is seen that the total number of rules is greatly reduced.

Remark 4. The highlighted terms in Table 5 are for increasing the mapping variables, such as the mapping variables of I, J, and K in Table 11. Without the highlighted terms, the mapping variable will be only one, as one variable I in Table 11. This will reduce the rules of F_5 in Table 10.

Example 2. In a seesaw system as shown in Fig. 10, let \(x\) be the distance of the cart from the origin, \(\theta\) be the angle that the wedge makes with the vertical line, \(r_1\) be the height of the wedge and \(r_2\) be the center of mass of the wedge.
The system can be represented by the dynamical equation as

\[ u + mg \sin \theta = m \ddot{x}, \]

\[ (Mg \sin \theta) \cdot r_2 + mg \sin(\theta + \phi) \cdot \sqrt{x^2 + r_1^2} + ur_1 = I \ddot{\theta}, \]

where \( I \) is the wedge inertia, \( \phi \) is the angle that the cart makes with the wedge vertical line and the system parameters \((m, M, r_1, r_2, I)\) are \((0.46, 1.41, 0.143, 0.121, 0.095)\).

The structure of HFS is shown in Fig. 11, and, Table 12 is the symmetrical control rule base of the single layer fuzzy logic system constructed by the following if–then forms:

- if \( x = pb, \dot{x} = pb, \theta = pb, \dot{\theta} = pb \), then \( u = nb \),
- if \( x = pb, \dot{x} = pb, \theta = pb, \dot{\theta} = ze \), then \( u = nb \),

... 
- if \( x = nb, \dot{x} = pb, \theta = nb, \dot{\theta} = nb \), then \( u = pb \).

We herein use the algorithm for L-HFS to obtain the \( F_x \), \( F_\theta \) and \( F_u \) as shown in Tables 13 to 15, respectively. The controller uses the triangular membership function for the input variables, the singleton for the output variables, which are as shown in Fig. 12. The method of fuzzy inference is Mamdani’s. By using L-HFS, the total number of rules is 43, that means, there are 9 rules for \( F_x \), 9 rules for \( F_\theta \), and 25 rules for \( F_u \). It is easily seen that the rules are reduced when compared with conventional single layer fuzzy logic system, whose rules are 81 \((3 \times 3 \times 3 \times 3 = 81)\).
Table 12
The rules of the single layer

<table>
<thead>
<tr>
<th>( \dot{\theta} )</th>
<th>( \dot{x} )</th>
<th>( x )</th>
<th>( \dot{x} )</th>
<th>( x )</th>
<th>( \dot{x} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>ze</td>
<td>ze</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>nb</td>
<td>pb</td>
<td>nb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>nb</td>
<td>pb</td>
<td>nb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>nb</td>
<td>pb</td>
<td>nb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
</tbody>
</table>

Table 13
The rules of the \( F_x \)

<table>
<thead>
<tr>
<th>( \dot{x} )</th>
<th>( x )</th>
<th>( \dot{x} )</th>
<th>( x )</th>
<th>( \dot{x} )</th>
<th>( x )</th>
</tr>
</thead>
<tbody>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>ze</td>
<td>ze</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
</tbody>
</table>

Table 14
The rules of the \( F_\theta \)

<table>
<thead>
<tr>
<th>( \dot{\theta} )</th>
<th>( \dot{x} )</th>
<th>( x )</th>
<th>( \dot{x} )</th>
<th>( x )</th>
<th>( \dot{x} )</th>
<th>( x )</th>
</tr>
</thead>
<tbody>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>ze</td>
<td>ze</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
</tbody>
</table>

Table 15
The rules of the \( F_u \)

<table>
<thead>
<tr>
<th>( u_\theta )</th>
<th>( u_x )</th>
<th>( A )</th>
<th>( B )</th>
<th>( C )</th>
<th>( D )</th>
<th>( E )</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>nb</td>
<td>nb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>G</td>
<td>nb</td>
<td>nb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>H</td>
<td>nb</td>
<td>nb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>I</td>
<td>nb</td>
<td>nb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
<tr>
<td>J</td>
<td>nb</td>
<td>nb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
<td>pb</td>
</tr>
</tbody>
</table>
Fig. 12. The membership function of input and output variables.

Fig. 13. The simulation result of seesaw system with a single layer fuzzy system.

Figs. 13 and 14 show the simulated results of the control of a seesaw system with the conventional single layer fuzzy logic system and the L-HFS during 10s from the initial conditions with \((x \dot{x} \theta \dot{\theta})\) being \((0.2, 0, 10, 0)\). And the scaling factors, shown in Fig. 11, are chosen by self-tuning GA, with \((k_x k_\dot{x} k_\theta k_\dot{\theta})\) being \((0.2066, 0.002, 0.2023, 0.0374, 711.5)\). The simulated results show that we can balance the seesaw system by using the proposed mapping rules and the same scaling factors.

**Remark 5.** In order to choose the scaling factors, we can utilize the self-tuning GA with the following fitness function defined as

\[
f_{objective} = (1 + k_1 \times GRT)(1 + k_2 \times GOV)(1 + k_3 \times GSSE)(1 + k_4 \times GST),
\]

where

\[
GRT = \exp(-RT/[RT]_{expected}),
\]

\[
GOV = \exp(-OV/[OV]_{expected}),
\]

\[
GSSE = \exp(-SSE/[SSE]_{expected}),
\]

\[
GST = \exp(-ST/[ST]_{expected}),
\]
and $RT$ is the rising time, $OV$ is the overshoot, $SSE$ is the steady-state error, and $ST$ is the settling time. The constants $k_1$–$k_4$ are used as the weighting of the corresponding fitness function. The values of $[RT]_{\text{expected}}$, $[OV]_{\text{expected}}$, $[SSE]_{\text{expected}}$ and $[ST]_{\text{expected}}$ are the acceptable specifications of the rising time, the overshoot, the steady-state error, and the settling time, respectively.

**Remark 6.** In fact, if the input/output mapping is the same, the system response will be affected by a great number of rules reduced. This phenomenon can appear from Figs. 13 and 14, a system response with single layer fuzzy system and with proposed HFS. Nevertheless, by using the new method, the system stability can be ensured as shown in Fig. 14.

6. Reducing the number of rules

In this section, we will discuss the effect of the proposed method in terms of reducing the number of rules. Basically, because the HFS structures of Figs. 15 and 7 are equal, they can have the same input–output model before and after the hierarchical decomposition. For the convenience of description, the HFS structure of Fig. 15 is used for the discussion about reducing the number of rules. In Fig. 15, the numbers of membership functions for each state are $n_1, n_2, \ldots, n_n$, respectively, and the numbers of mapping variables in each layer is $m_1, m_2, \ldots, m_{n-2}$, respectively. The maximum number of mapping variables in each layer is as follows:

\[
m_1 : n_1 \cdot n_2,
\]

\[
m_2 : n_1 \cdot n_2 \cdot n_3,
\]

\[
\vdots
\]

\[
m_{n-2} : n_1 \cdot n_2 \ldots n_{n-1}.
\]
The total number of rules of the HFS sum up to total 1: \( n_1 \cdot n_2 + m_1 \cdot n_3 + m_2 \cdot n_4 + \cdots + m_{n-2} \cdot n_n \).

But, the total number of rules in the single layer fuzzy system sum up to total 2: \( n_1 \cdot n_2 \cdot n_3 \cdots n_n \).

If one wants to reduce the number of involved rules, then it is required that \((\text{total 1} - \text{total 2}) < 0\).

In other words, the numbers of mapping variables used must satisfy this condition.

7. Conclusion

A new kind of the mapping rule base scheme of hierarchical fuzzy systems was proposed. The present method can reduce the number of involved rules and can overcome the difficulty of determination of scaling factors of HFSs. The method is shown to be effective and can be extended to higher dimensional HFSs, and it can be found that the number of rules will be greatly reduced for a six-variable input system as described in this work. However, if too many mapping variables are used, such as, for a three-variable input system as shown in Fig. 5, if there are over six mapping variables for the output of \( u_0 \), then the total number of rules will be over 27, that means there are 9 rules to get the output of \( u_0 \) and over 18 rules to get the output of \( u \). Therefore, the number of involved rules may not decrease. But, in the usual case, we have a symmetrical control rule base, which has a regular pattern, and it only uses several mapping variables as well as satisfies the condition in the previous section.

Acknowledgements

The authors wish to thank the reviewers for their constructive comments which have made substantial improvement in this paper.
In addition, the authors also appreciate the financial support of the National Science Council of the Republic of China under Contract NSC 91-2213-E008-023.

References