Proceedings of
NATIONAL SYMPOSIUM
on
COMPUTER APPLICATIONS
in
POWER PLANTS

December 8-10, 1992

Bhabha Atomic Research Centre
Bombay 400 085

Sponsored by

Board of Research in Nuclear Sciences
DAE
Abstract

In this paper, we present the design of an Advanced Computer which provides an Expert System for use in Automatic Control Applications (e.g. Automation of Power Plants in Nuclear, Thermal and Electrical Industries). The Inference Engine of this Expert System relies on Fuzzy Logic for its Intelligence. We discuss algorithms along with the supporting structures and relations for the construction of the Inference Engine. The algorithms are parallelized and suitable for SIMD architectures.

1. Introduction

Successful applications based on Fuzzy Logic are being announced by Industry with a regularity that ratifies the rationale that Fuzzy Logic-based Advanced Computing Systems are indeed the machines of the future. Most of the applications involve some form of automatic control or other. Automation of Power Plants (Nuclear, Thermal, Cement etc..), Vehicular automation, Domestic appliance control etc.. are some typical areas that are using solutions based on Fuzzy Logic.

1.1 Features of FUZZY Logic-based control

Some features of Fuzzy Logic-based Controllers (FLCs) that have played a major role in their popularity are:

(a) FLCs use a rule-base containing rules derived through basic intuition and/or through talks with human experts.

(b) Outputs of FLCs can adapt automatically to changes in input parameters.

(c) FLCs can non-linearities in inputs since the manipulations required to generate an O/P are themselves based on curves.

(d) The output of a FLC is very smooth and due to this
life-span of the controlled equipment increases and its energy-consumption is highly optimized.

(e) A Fuzzy Logic-based system is intrinsically robust and Fault-tolerant since every rule in the rule-base of the system is used to arrive at final control action. Hence, the system continues to function even if some of input devices (sensors) fail.

(f) The accuracy and the precision of the output of the FLC far exceeds what can be obtained from a conventional controller or a human operator [1].

(g) The User Interface to the system is a true Natural Language Interface as the system is programmed in plain English. Above features are better exploited by encapsulating them in an advanced architecture modeled on a Fuzzy Expert System.

2. Basic Concepts

In Fuzzy Logic, all variables have their values mapped elastically into a curve representing the Linguistic term associated with the variable. The result is called a Fuzzy Set and it has a gradation called the Membership Value of the Linguistic variable, actually the ordinate, in the range [0,1]. The curves have a domain of [-1,1].

Each Fuzzy rule has many antecedents (input variables) and one consequent (output variable). The membership values obtained from the antecedents are combined into a single value called the premise using Fuzzy Logic principles - an AND combination gives the minimum of the values, an OR gives the maximum etc.

3. Functional description

We propose the following architecture for the Fuzzy Logic-based advanced computer (Figure 1.1).

![Diagram](image)

**FIGURE 1.1**

It is an Expert System with appropriate modifications. The
knowledge-base contains production-rules of IF/THEN type and is essentially built up by actually talking to the human expert on the controlled device, referring operating manuals of the device or by sheer intuition. Hence, the knowledge-base is partly responsible for the human-like attributes of the Fuzzy system.

The Fuzzy computer can be programmed in plain English; programming comprises of knowledge-gathering and/or tuning the system to yield desired output levels. The knowledge is captured using a well-defined User Interface and a Feedback path is provided for tuning purposes. The knowledge-base section incorporates data structures related to the Fuzzy Membership curves and it could possibly have an object-oriented memory architecture for storing its information.

The Inference Engine interprets the production-rules based on inputs received from the external world (through sensors etc..) and delivers a control output. The process by which this is done is dealt with in the following section (3.1.). The I/O section is similar to that of a conventional computer.

3.1. The FUZZY inference engine

The Inference Engine of a Fuzzy Expert System operates on production-rules and method of evaluating these rules (rule-firing) corresponds to logical inference method of modus ponens. This method is data-driven - all available data is supplied to Expert System, which then uses it to evaluate production-rules and draws conclusions resulting in the output control action.

3.1.1. Inferencing and defuzzification

The Fuzzy rules fire to an extent determined by the premises' truth value in contrast with conventional logic where rules fire 100% or not at all. Thus, solving a rule determines the portion of the output variable - an area under the curve - that is used along with similar areas from every other rule in the rule-base (pertaining to the same output variable) to find the concluding output control action. This process is called Inferencing. Thus, Inferencing transfers the premise truth value to the consequent. Of the many methods available for Inferencing, we use the MAX-DOT method which scales the membership curves of the Fuzzy output set to peak at the corresponding premise truth value (also referred to as the Scaling Factor).

The scaled output variables (areas under curves) are then graphically added to create a composite Fuzzy set. The process of conversion of this set to a real number is called Defuzzification. The most common method to do so involves obtaining the abscissa of the centroid of the area under the curve representing the composite Fuzzy set.
3.1.2. Algorithms for the FUZZY inference engine

We now discuss the algorithms developed by us to implement the Fuzzy Inference Engine.

**Initialization:** This phase involves establishment of production-rules, identification of Linguistic variables and Fuzzy modifiers for use in the rules, definition & design (mostly though intuitive or learning/tuning mechanism) of the Fuzzy curves associated with the input & output variables of the rules. We recommend the use of parabolic sections for construction of smooth S- or π-Curves for representing Fuzzy sets [3] as they simplify calculations and at the same time yield fine-grained output control. If parametric equations are used to represent the parabolic sections, then tuning of the Inference Engine reduces to adjusting these parameters so as to get the desired shape and/or output action. Cross-over points represent the transitions from one section of a curve to an adjoining section.

Most of the parts of the Fuzzy curve database shown in Figure 1.2 are calculated apriori in the initialization phase so that computation during actual operation is reduced and faster. Every curve has a unique identification number. We assume that each curve is made up of a maximum of 4 sections and that as far as possible, any two curves intersect in only one point apart from common range limits, if any.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Curve Identification Number.</td>
<td>2 - Number of Sections making up the curve.</td>
<td>3 - Range Limits for each Section.</td>
<td>4 - Parameters of the Parabolas forming the Sections (Coordinates of vertices and lengths of latus rectums).</td>
<td>5 - Unscaled Intersection-points of this curve with all other curves associated with the same Output Variable.</td>
<td>6 - Area under the curve &amp; its Centroid.</td>
<td>7 - Threshold Scaling Factor Sets - One Set per Curve-pair; each Set having TSF I and TSF II as elements.</td>
<td>8 - Pointers to routines associated with computations related to this curve.</td>
</tr>
</tbody>
</table>

**FUZZY CURVE DATABASE**

Figure 1.2

It may be noted that this database can be effectively represented by Object-oriented techniques.

Area under a curve is calculated by section-wise and Simpson's rule. Centroids of areas under unscaled curves are also obtained similarly [4]. **Threshold Scaling Factors** (TSFs) are also calculated apriori. A TSF is that value of a scaling factor (or
truth value of the premise of a rule) which scales down the curve involved to such an extent that the intersection-point (IP) of that curve Figure 1.1 with another slides over from one parabolic section of the curve an adjoining section; this implies that the parameters of that new section have to be used for further calculations. The selection of a suitable cross-over point can be determined in terms of its proximity to the original IP and using the slope of the line joining them. Every curve-pair has associated with it a set of 2 TSFs because when 2 intersecting curves are scaled-down, we determine the new IP by using an intermediate IP which is got by scaling-down one curve and keeping the other unscaled; later the other curve is also scaled-down to effect the actual intersection. TSF-I is used for determining the coordinates of the intermediate IP from the original (unscaled) IP and TSF-II is used to get the new IP from the intermediate values. TSFs are computed only for curves occurring in consequent parts of rules.

Inferencing: Assume that N rules in rule-base are associated with the output control variable that we desire to control. Our algorithm exploits the inherent parallelism in Fuzzy Logic principles and is an SIMD parallel scheme and uses the parallel construct PAR which essentially causes the machine to concurrently execute the statements that follow the directive.

SIMD (Single Instruction - Multiple Data) Algorithm)

SEQ

PAR /* For every concerned rule. */
PAR /* For each input variable of a rule. */
SEQ
OBTAIN EXTERNAL INPUT;
NORMALIZE EXTERNAL INPUT;
OBTAIN MEMBERSHIP-VALUE IN FUZZY SET;
OBTAIN PREMISE (SCALING FACTOR) FOR RULE;
PAR /* For every rule-pair (i,j) s.t i <> j, i < j. */
SEQ
USE TSF OF j w.r.t i AND UNSCALED IP TO GET INTERMEDIATE IP;
/* Keep i fixed and scale down j */
USE TSF OF i w.r.t j AND INTERMEDIATE IP TO GET NEW IP;
/* Keep j fixed and scale down i */
COMPUTE INTERSECTION-AREA AND ITS CENTROID USING NEW IP;
COMPUTE WHOLE AREAS UNDER SCALED-DOWN CURVES i, j & THEIR CENTROIDS;
ADD UP ALL WHOLE-AREAS AND SUBTRACT ALL INTERSECTION-AREAS TO GET EFFECTIVE-AREA;
COMPUTE CENTROID OF EFFECTIVE-AREA;
DE-NORMALIZE ABCISSA OF CENTROID OF EFFECTIVE-AREA & OUTPUT FINAL, DEFUZZIFIED CONTROL ACTION TO DEVICE;

Supporting relations: Normalization converts an external
input from a range \((a,b)\) to a range \([-1,1]\) for mapping into Fuzzy Curves and De-normalization converts back the truth value of the premise \((0,1)\) into \((a,b)\). The relation used is:
\[
x_n = \frac{2x_{in} - (b+a)}{b-a}
\]
where \(x_n\) is normalized value of \(x_{in}\).

De-normalization can be done by inverting the above relation.

TSFs are calculated for 2 curves \(i\) and \(j\) with vertices at \((h_i,k_i)\) and \((h_j,k_j)\) & latus rectums of lengths \(p_i\) and \(p_j\) respectively, using:
\[
\begin{align*}
\text{TSF-I} &= 1 + \frac{\left((2b_i-h_i) - (h_i-x_{\text{orig}})\right)^2}{4p_iy_{\text{orig}}} \\
\text{TSF-II} &= 1 + \frac{\left((2b_j-h_j) - (h_j-x_{\text{inter}})\right)^2}{4p_jy_{\text{inter}}}
\end{align*}
\]
(Keep curve \(i\) unscaled and scale curve \(j\))

(Keep curve \(j\) unscaled and scale curve \(i\))

where, Original (unscaled) IP is at \((x_{\text{orig}},y_{\text{orig}})\) and the intermediate IP is at \((x_{\text{inter}},y_{\text{inter}})\).

4. Conclusion

We have discussed SIMD algorithms and techniques for implementing the Inference Engine of a Fuzzy Logic-based advanced computing system. These are highly parallel intrinsically and their speed of execution can be enhanced by upgrading them to a MIMD (Multiple Instruction, Multiple Data) architecture. This aspect gets rid of the time problems associated with the computation of the complex analytical manipulations involved. The use of Object-oriented techniques will also be helpful.

Since the resulting Expert System is easy to program and yields finer outputs, its utility in automatic control is immense.

5. References

[1]: FUZZY logic is anything but fuzzy - Tom Williams; Computer Design; pp. 113-127; Apr 1992.
[3]: FUZZY sets and their applications to cognitive and decisive processes - Edited by L.A.Zadeh et al; Academic Press; 1975.
[4]: Calculus with analytical geometry - Johnson and Kiokemeister; Prentice Hall of India.