Stochastic fuzzy controller

Franc Jurkovič

A standard approach to building a fuzzy controller based on stochastic logic uses binary random signals with an average (expected value of a random variable) in the range [0, 1]. A different approach is presented, founded on a representation of the membership functions with the probability density functions.

Introduction: As we know, an implementation of fuzzy inference can be done on general purpose computer architectures (including digital signal processors and microcontrollers) or on a special fuzzy hardware [1]. Analogous and digital techniques are used. Fuzzy controllers working on stochastic logic represent a special kind of fuzzy hardware [2], [3]. The stochastic logic uses binary random signals with an average (mean) value in the interval between 0 and 1. It saves procedures known in the analogous technique, but it need digital compounds for its realization. It is possible to obtain a very simple realization of multiplication. The ways of signal representation are different. Synchronous or asynchronous working and various modulations are possible [4]. Computation with random signals has some advantages in comparison with classical techniques. Stochastic signals are much less sensitive to noise than analogous signals. Digital logic can be very space-efficient.

Results: The fuzzy logic controller is described as the fuzzy logic system [5]. Its task is to map a sharp input value into a sharp output one. A fuzzyficator arranges the corresponding input fuzzy sets to sharp input values. An inference mechanism maps the input fuzzy sets into output fuzzy sets by using linguistic rules. Further, a defuzzyficator transforms them into a sharp output value. In the case of two input values, the linguistic rules have the following form:

$$
\begin{aligned}
&\text{if} \quad x_1 = A_1 \quad \text{and} \quad x_2 = B_1 \quad \text{then} \quad y = C_1 \\
&\text{if} \quad x_1 = A_2 \quad \text{and} \quad x_2 = B_2 \quad \text{then} \quad y = C_2 \\
&\ldots \\
&\text{if} \quad x_1 = A_i \quad \text{and} \quad x_2 = B_i \quad \text{then} \quad y = C_i \\
&\ldots \\
&\text{if} \quad x_1 = A_m \quad \text{and} \quad x_2 = B_m \quad \text{then} \quad y = C_m
\end{aligned}
\tag{1}
$$

or described by implication:

$$
\begin{aligned}
A_1 \times B_1 &\Rightarrow C_1 & = & \quad R_1 \\
A_2 \times B_2 &\Rightarrow C_2 & = & \quad R_2 \\
&\dots \\
A_i \times B_i &\Rightarrow Ci & = & \quad R_i \\
&\dots \\
A_m \times B_m &\Rightarrow C_m & = & \quad R_m
\end{aligned}
\tag{2}
$$

We unify rules into a single relation [6]:

$$
R = \cup R_i
\tag{3}
$$

For inference operator, the Larsen product implication is used:

$$
C' = (A' \times B') \cdot R
\tag{4}
$$

C' is an output fuzzy set, that is a mapping of input fuzzy sets A' and B' with consideration of the rules included in the relation R. The problem of computation output fuzzy sets C' passes over to fuzzy sets (the membership functions) multiplication. By consideration of the principle "function is a function" we reach an analogy between fuzzy and probability:

$$
\mu(x_i) \Leftrightarrow P\{X = x_i\}
\tag{5}
$$

The analogy with the membership function $\mu(x_i)$ is the probability density function $P\{X = x_i\}$ of the random variable X.

Only simultaneous equal realizations of both random variables that represent functions μ_1 and μ_2 contribute to forming the resulting probability density function $\mu_1 \cdot \mu_2$ Defuzzyfication by the COG (center of gravity) method has an analogy in the computation of the expected value E of the random variable:

$$
y_o = \sum_i y_i \cdot \mu(y_i) \Leftrightarrow E(Y) = \sum_i y_i \cdot P\{Y = y_i\}
\tag{6}
$$

since the probability sum of single realizations of the random variable is 1.

By derivation we have reached a point when all theoretical resources, needed in controller design, are ready. An implementation requires [7] a realization of multiplication of the probability density functions, a realization of their generation, and a realization of defuzzyfication.

Example: Fig. 1 displays a block scheme of the controller. The shift register SR with the starting logic SL and with feedback binding over the XOR2 gate and input, connected with the XOR1 gate, generates pseudo-random numbers with the uniform distribution. The adders ADD1, ADD2, and ADD3 with added delays D1, D2, and D3, respectively, add three times two serial pseudo-random numbers, and the approximated triangular distributions of pseudo-random numbers is obtained on the output. They are statistic independent. To realize the membership functions S, M, and B shifts C1 and C2 are needed respectively. On its input, the multiplexer MUX (creation of rule union) has attached combinations of the membership function corresponding to singular controller rules. Rule 1:

$$\text{if} \quad XA = S(\text{small}) \quad \text{and} \quad XB = S(\text{small}) \quad \text{then} \quad Y = B(\text{big}) \tag{7}$$

Signals *XA* and *XB* on the multiplexer output representing the causal part of a rule compared with input values *XA'* and *XB'* in the comparator COM. At the times when equality is reached, the comparator connects with the switch SW the signal Y that on multiplexer output represents the consecutive rule part, to the low-pass filter NF through the digital-analog converter (D/A). The analog-digital converters (A/D) convert the input analog values Xa and Xb into digital values XA' and XB'.

A disadvantage of the described controller is its relative slowness (bad dynamic properties). A fixed time is needed for the output signal y_o to be formed. We should not to forget that this is a statistic property (expected value of random signal). Simulation results have shown the regularity of the new concept, and for the possible realization, e.g. in the FPGA (field programmable gate arrays) technique, there are no greater obstacles.

References:

[1] A. KANDEL, AND G. LANGHOLZ, ed., *Fuzzy hardware, architectures and applications*, Kluwer Academic Publishers, Boston, London, Dordrecht, 1998.

[2] A. TORRALBA, F. COLODRO, AND L.G. FRANQUELO, "A fuzzy-logic controller with on-chip learning, employing stochastic logic", in *Proc. Of the 3rd IEEE Int. Conference on Fuzzy Systems*,

Orlando, USA, June 1994, pp. 1759-1764.

[3] F. COLODRO, A. TORRALBA, R. CARVAJAL, AND L.G. FRANQUELO, "A fuzzy-logic controller using stochastic logic", in *Proc. Of the 1997 IEEE Int. Symp. On Circuits and Systems*, Hong Kong, June 1997, pp. 629-632.

[4] L.M. REYNERI, "A performance analysis of pulse stream neural and fuzzy computing systems", *IEEE Trans. Circuits and Systems – II: Analog and digital signal processing*, vol.42, no.10, 1995, pp. 642-660.

[5] J.M. MENDEL, "Fuzzy logic systems for engineering: a tutorial", *Proceedings of the IEEE*, vol.83, no.3, March 1995, pp. 345-377.

[6] W. PEDRYCZ, "An approach to the analysis of fuzzy systems", *Int. J. Control*, vol.34, no.3, 1981, pp. 403-421.

[7] G.A. KORN, *Random-process simulation and measurements*, McGraw-Hill, New York, 1966.

Authors' affiliation:

Franc Jurkovič

University of Maribor, Faculty of Electrical Engineering and Computer Science

Smetanova 17, SI-2000 Maribor

Slovenia

e-mail: franc.jurkovic@uni-mb.si

Figure captions:

Fig. 1: Block scheme of the controller example