Simulated Annealing for Automated Definition of Fuzzy Sets in Human Central Nervous System Modeling

Francisco Mugica[1] and Angela Nebot[2]

[1] Instituto Latinoamericano de la Comunicación Educativa
Calle del Puente, 45, Col.Ejidos de Huipulco, México D.F. 14380, México
fmugica@ilce.edu.mx
[2] Dept. Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya
Jordi Girona Salgado, 1-3, Barcelona 08034, Spain
angela@lsi.upc.es

Abstract. The main goal of this research is to study the usefulness of the Simulated Annealing (SA) approach, developed in the context of the Fuzzy Inductive Reasoning (FIR) methodology, for the automatic identification of fuzzy partitions in the human Central Nervous System (CNS) modeling problem. The SA algorithm can be viewed as a preprocess of the FIR methodology that allows the modeler to use it in a more efficient way. Two different SA algorithm cost functions have been studied and evaluated in this paper. The new approach is applied to obtain accurate models for the five controllers that compose the CNS. The results are compared and discussed with those obtained by other inductive methodologies for the same problem.

1 Introduction

The human central nervous system controls the hemodynamical system, by generating the regulating signals for the blood vessels and the heart. These signals are transmitted through bundles of sympathetic and parasympathetic nerves, producing stimuli in the corresponding organs and other body parts.

In this work, CNS controller models are identified for a specific patient by means of the Fuzzy Inductive Reasoning (FIR) methodology. FIR is a data driven methodology that uses fuzzy and pattern recognition techniques to infer system models and to predict its future behavior. It has the ability to describe systems that cannot easily be described by classical mathematics (e.g. linear regression, differential equations) i.e. systems for which the underlying physical laws are not well understood. The FIR methodology is composed of four main processes, namely: *fuzzification*, *qualitative model identification*, *fuzzy forecast* and *defuzzification*.

The first step of the FIR methodology is the *fuzzification* process, that converts quantitative data stemming from the system into fuzzy data. In this process the number of classes of each variable (i.e. the partition) needs to be provided.

A. Sanfeliu and J. Ruiz-Shulcloper (Eds.): CIARP 2003, LNCS 2905, pp. 545–553, 2003.

In this paper an algorithm based on a simulated annealing technique/method, developed in the context of FIR, is used to automatically suggest a good partition of the system variables in an efficient way. The SA algorithm can be viewed as a pre-process of the FIR methodology that allows the modeler not to rely on heuristics for the definition of a system variable partition. Two SA algorithm cost functions are proposed in this research that make use of the qualitative model identification and the forecast processes of FIR methodology. A brief description of these processes are given next. The *qualitative model identification* process of the FIR methodology is the responsible to find causal and temporal relations between variables and therefore to obtain the best model that represents the system. A simplified diagram of the qualitative model identification process is presented in figure 1.

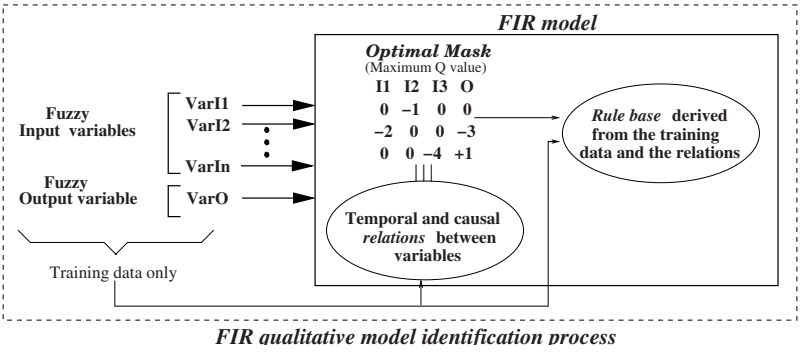

Fig. 1. Simplified diagram of the FIR qualitative model identification process

A FIR model is composed of a mask (model structure) and a pattern rule base. An example of a mask is presented in figure 1. Each negative element in the mask is called a m-input (mask input). It denotes a causal relation with the output, i.e. it influences the output up to a certain degree. The enumeration of the m-inputs is immaterial and has no relevance. The single positive value denotes the output. In position notation the mask of figure 1 can be written as $(2, 5, 8, 11, 12)$, enumerating the mask cells from top to bottom and from left to right. The qualitative identification process evaluates all the possible masks and concludes which one has the highest prediction power by means of an entropy reduction measure, called the quality of the mask Q. The mask with the maximum Q value is the optimal mask. Starting from the fuzzified system data and using the optimal mask, the pattern rule base is then synthesized. Both, the pattern rule base and the mask constitute the FIR model. Once the pattern rule base and the optimal mask are available, system predictions can take place using FIR inference engine. This process is called *fuzzy forecast*. FIR inference engine is a specialization of the k-nearest neighbor rule, commonly used in the pattern recognition field. *Defuzzification* is the inverse process of fuzzification. It

allows to convert the qualitative predicted output into quantitative values that can then be used as inputs to an external quantitative model. For a deeper inside of the FIR methodology refer to [1].

2 Simulated Annealing for Identification of Fuzzy Partitions in FIR

Simulated annealing is a generalization of a Monte Carlo method and it is used to approximate the solution of large combinatorial optimization problems [4]. A simulated annealing algorithm consists of two loops. The outer-most loop sets the temperature and the inner-most loop runs a Metropolis Monte Carlo simulation at that temperature. The algorithm starts with an initial solution to the problem, which is also the best solution so far and a value for an initial high temperature. Each iteration consists of the random selection of a new solution (candidate solution) from the neighborhood of the current one. The cost function of the candidate solution is evaluated and the difference with respect to the cost function value of the current solution is computed. If this difference is negative the candidate solution is accepted. If the difference is positive the candidate solution is accepted with a probability based on the Boltzmann distribution. The accepted candidate solution becomes the current solution and if its cost function value is lower than the one of the best solution, this one is updated. If the candidate solution is rejected the current solution stays the same and it is used in the next iteration. The temperature is lowered in each iteration down to a *freezing* temperature where no further changes occur. A detailed description of the simulated annealing algorithm developed for the automatic identification of fuzzy partitions in the FIR methodology can be found in [2].

Two main aspects of the simulated annealing algorithm that need to be considered here are the *new solution generation mechanism* and the *cost function*. Both are highly important to achieve a good performance of the algorithm.

The *new solution generation mechanism* consists of two tasks. The first one is the generation of the initial partition at the beginning of the algorithm execution. The second one is the generation of a new solution (i.e. candidate solution) starting from the current solution, in each algorithm iteration. Two options have been studied in this paper to generate an initial partition: 3-classes partition and random partition. The first one sets all the variables to 3 classes. The second one performs a random generation of the number of classes for each system variable. In this research the number of classes allowed for each system variable is in the range $[2 \ldots 9]$.

The procedure to generate a new solution, i.e., the candidate solution, from the current one is to increment or decrement by one the number of classes associated to a certain system variable. The variable that is going to be modified is chosen randomly out of the vector of variables. The decision to increase or decrease the number of classes of this variable is also randomly taken.

Two different *cost functions* have been studied in this work: the quality of the optimal mask and the prediction error of the training data set.

As has been explained earlier, in the *qualitative model identification* process of the FIR methodology the optimal mask (i.e. the best model structure) is identified by means of a quality measure, Q. The quality of a mask is a value between 0 and 1, where 1 indicates the highest quality. Therefore, the first cost function proposed is $1 - Q$, due to the fact that the algorithm task should minimize the cost function.

The second cost function is defined as the prediction error of a portion of the training data set. The normalized mean square error in percentage (MSE), given in equation 1, is used for this purpose.

$$MSE = \frac{E[(y(t) - \hat{y}(t))^2]}{y_{\text{var}}} \cdot 100\% \tag{1}$$

$\hat{y}(t)$ is the predicted output, $y(t)$ the system output and y_{var} denotes the variance of $y(t)$. The idea is to use part of the training data set to identify the model and the rest of the data set to evaluate the prediction performance of that model. The prediction error of the portion of the training data set not used in the model identification process is used as the cost function for the SA algorithm. The size of the portion of the training data set actually used for cost function evaluation purposes is defined with respect to the size of the whole training data set.

3 Central Nervous System Modeling

The central nervous system is composed of five controllers, namely, *heart rate* (HR), *peripheric resistance* (PR), *myocardiac contractility* (MC), *venous tone* (VT) and *coronary resistance* (CR). All the CNS controllers are SISO models driven by the same input variable, the *carotid sinus pressure* (CSP). The input and output signals of the CNS controllers were recorded with a sampling rate of 0.12 seconds from simulations of the purely differential equation model [3], obtaining 7279 data points. The model had been tuned to represent a specific patient suffering a coronary arterial obstruction, by making the four different physiological variables (right auricular pressure, aortic pressure, coronary blood flow, and heart rate) of the simulation model agree with the measurement data taken from the real patient. The five models obtained were validated by using them to forecast six data sets not employed in the training process. Each one of these six test data sets, with a size of about 600 data points each, contains signals representing specific morphologies, allowing the validation of the model for different system behaviors.

The main goal of this research is to study the usefulness of the SA approach as a pre-processing tool of the FIR methodology for the identification of good models for each of the five controllers. Let us explain the experimentation procedure for the coronary resistance controller. The same strategy has been used for the other four controllers. Their results are presented later.

As mentioned before, two cost functions were studied in this work. Table 1 shows the results obtained for the coronary resistance controller when $1-Q$ was used as cost function. Table 2 presents the results of the same controller when

Table 1. Partition results of the CR controller obtained using $1 - Q$ as cost function

Ini.Part. CSP CR	Fin.Part. CSP CR	Opt.Mask	Q	$1 - Q$	MSE_{test}	#GS	Time
(3,3)	(9,3)	(1,4,6)	0.9787	0.0213	3.85%	35	2.98
(3,3)	(7,3)	(1,4,6)	0.9776	0.0224	4.76%	26	1.97
(3,3)	(8,3)	(1,4,6)	0.9776	0.0224	4.25%	37	2.56
(3,3)	(6,3)	(1,4,6)	0.9762	0.0238	1.75%	27	1.41
(3,3)	(5,3)	(1,4,6)	0.9749	0.0251	2.34%	24	1.69
(3,3)	(4,3)	(1,4,6)	0.9748	0.0252	1.33%	26	1.38
(8,8)	(9,3)	(1,4,6)	0.9787	0.0213	3.85%	33	6.17
(9,6)	(9,3)	(1,4,6)	0.9787	0.0213	3.85%	26	3.18
(7,2)	(9,3)	(1,4,6)	0.9787	0.0213	3.85%	35	2.38
(6,5)	(9,3)	(1,4,6)	0.9787	0.0213	3.85%	25	3.60
(5,5)	(9,3)	(1,4,6)	0.9787	0.0213	3.85%	35	3.24
(5,2)	(5,3)	(1,4,6)	0.9749	0.0251	2.34%	37	1.44
	Optimal	Solution:	Opt.Mask=	(9,3);	Q=	0.9787;	

the cost function is defined as the prediction MSE of a portion of the training data set. In this application the last 25% of the training signal is used for cost function evaluation and only the first 75% of the signal is used to obtain the FIR models.

Both, the 3-classes and the random options have been evaluated as initial partitions. The upper rows of tables 1 and 2 show the results of the 3-classes initial partition, whereas the lower rows present the results of the random initial partition. For both options, 40 executions of the SA algorithm were performed. For an initial partition of 3 classes the SA algorithm suggested up to 6 different final partitions when $1 - Q$ is used as cost function (see table 1) and 3 possible final partitions when the prediction error is used as cost function (see table 2). When the random initial partition is used, only 2 and 4 different final partitions are suggested by the SA algorithm for the $1 - Q$ and prediction error cost functions, respectively.

The tables are organized as follows. The first column indicates the initial partition from which the SA algorithm starts the search. The second column presents the final partition suggested by the SA algorithm when the cooler temperature is reached (i.e. the algorithm stops). Note that the final partition is the input parameter to the *fuzzification* process of the FIR methodology. The third and fourth columns contain the optimal mask obtained by FIR for that specific partition (in position notation) and its associated quality, respectively. The fifth column corresponds to the cost function evaluation. Note that in table 1 the cost function is $1 - Q$ and in table 2 the cost function is the prediction MSE of the last 25% data points of the training set. The next column shows the prediction error of the test data sets. As mentioned before, six test data sets of 600 data points each are available for each controller. The results presented in the tables are the mean value of the predictions errors obtained for these six test data sets. The seventh column indicates the total number of generated

Table 2. Partition results of the CR controller obtained using the prediction error of the last 25% of the training data set as cost function

Ini.Part. CSP CR	Fin.Part. CSP CR	Opt.Mask	Q	MSE_{train}	MSE_{test}	#GS	Time
(3,3)	(2,5)	(1,4,5,6)	0.9642	0.08%	0.15%	19	19.40
(3,3)	(3,4)	(4,5,6)	0.9638	0.12%	0.28%	32	38.39
(3,3)	(6,4)	(3,4,6)	0.9666	0.17%	0.42%	33	22.37
(4,4)	(2,5)	(1,4,5,6)	0.9642	0.08%	0.15%	24	17.51
(2,6)	(2,5)	(1,4,5,6)	0.9642	0.08%	0.15%	19	10.79
(3,5)	(3,4)	(4,5,6)	0.9638	0.12%	0.28%	23	12.31
(6,5)	(6,4)	(3,4,6)	0.9666	0.17%	0.42%	33	11.55
(5,4)	(6,4)	(3,4,6)	0.9666	0.17%	0.42%	30	16.28
(9,6)	(7,4)	(4,5,6)	0.9677	0.18%	0.41%	28	14.29
	Optimal	Solution:	Opt.Mask=	(2,5);	MSE_{train}= 0.08%;		

solutions during the execution of the SA algorithm. The last column contain the CPU time (in seconds) used by the algorithm to find the final partition. Clearly, the biomedical application presented in this paper is not a large optimization problem, it is rather small due to the fact that only two variables are involved and a maximum of nine classes is allowed (in fact there are only eight, because class 1 is not used). Therefore, there exists 64 possible solutions and an exhaustive search can be performed easily. However, it is interesting to work with a real application that shows clearly the usefulness of the SA algorithm for the automated definition of fuzzy sets in the FIR methodology. Moreover, the FIR performance is considerably increased when the SA algorithm is used in the CNS application.

If we look closer to table 1 it is clear that the optimal solution that corresponds to the $(9, 3)$ partition with a quality of 0.9787 is reached in both initial partition options. All the final partitions obtained when a $(3, 3)$ initial partition is used have in common that a partition of 3 classes is always suggested for the output variable, whereas 4,5,6,7,8 or 9 classes are good partitions for the input variable. Notice that the qualities of all the suggested partitions are very close to the optimal one. With a random initial partition, only two final partitions are suggested by the SA algorithm, i.e. the optimal one $(9, 3)$ and a suboptimal one $(5, 3)$. The proportion shown in table 1, i.e. five times partition $(9, 3)$ vs. one time partition $(5, 3)$ is the relation encountered in the 40 runs of the algorithm.

Table 3. MSE prediction errors of the CNS controller models using NARMAX, TDNN and RNN methodologies (mean value of the 6 test data sets for each controller)

	HR	**PR**	**MC**	**VT**	**CR**
NARMAX	9.3%	18.5%	22.0%	22.0%	25.5%
TDNN	15.3%	33.7%	34.0%	34.0%	55.6%
RNN	18.3%	31.1%	35.1%	34.7%	57.1%

Table 2 shows the results of the same controller when the prediction error of part of the training data set has been used as a cost function for the SA algorithm. The function to be minimized now is the MSE_{train}. It is interesting to remark, that in this case, the mask is obtained using exclusively the first 75% data points of the training signal. Therefore, the data used for the cost function evaluation has not been seen for the model before. This is the reason why the best predictions obtained for the last 25% values of the training set do not correspond necessarily to the partitions with the associated optimal mask of highest quality. However, the quality of the optimal masks found for the suggested partitions are still high, i.e. 0.96. The optimal solution is the partition $(2, 5)$ with a MSE_{train} of 0.08%, that is really very low. The SA algorithm is able to find the best final partition with both initial partition options, as happened also for the quality cost function. The $(3, 4)$ and $(6, 4)$ partitions with errors of 0.12% and 0.17%, respectively, are the best suboptimal solutions. Therefore, the SA algorithm obtains in fact the best three final partitions. Notice that although the number of generated solutions remains almost the same than table 1, the CPU time has considerably increased. This is due to the fact that the cost function evaluation is much more expensive computationally. Now, not only the qualitative model identification process of the FIR methodology is executed but also the fuzzy forecast process is.

Table 4. Partition results of the HR, PR, MC and VT controllers obtained using $1-Q$ cost function and prediction error of the last 25% of the training data set cost function

	HR			PR		
$1-Q$	Fin.Part.	$1-Q$	MSE_{test}	Fin.Part.	$1-Q$	MSE_{test}
	(7,2)*	0.1674	13.43%	(8,7)*	0.1448	5.99 %
	(8,2)	0.1861	12.63%	(7,7)	0.1505	4.59 %
	(7,4)	0.2739	2.61 %	(5,7)	0.1564	3.15 %
MSE_{train}	Fin.Part.	MSE_{train}	MSE_{test}	Fin.Part.	MSE_{train}	MSE_{test}
	(3,7)*	0.89%	9.15%	(4,9)*	0.93%	2.28%
	(5,9)	1.01%	2.54%	(7,7)	1.08%	3.34%
	(6,7)	1.15%	13.39%	(2,6)	1.64%	3.77%

	MC			VT		
$1-Q$	Fin.Part.	$1-Q$	MSE_{test}	Fin.Part.	$1-Q$	MSE_{test}
	(8,7)*	0.1866	11.88%	(8,7)*	0.1858	13.00%
	(7,7)	0.1950	42.45%	(7,7)	0.1952	41.88%
	(5,7)	0.2019	52.94%	(5,7)	0.2032	53.01%
MSE_{train}	Fin.Part.	MSE_{train}	MSE_{test}	Fin.Part.	MSE_{train}	MSE_{test}
	(4,9)*	0.60%	2.51%	(2,5)*	0.6117%	1.66%
	(2,5)	0.63%	2.74%	(2,8)	0.6359%	1.55%
	(3,9)	1.10%	3.87%	(3,7)	0.7855%	2.12%

It is interesting to analyze the MSE_{test} columns of both tables. As expected, the MSE_{train} cost function is able to obtain partitions with higher performance

on the prediction of the test data sets than the ones obtained by the $1-Q$ cost function. However, the results obtained in both cases are very good if compared with the ones obtained when other inductive methodologies are used . Table 3 contains the predictions achieved when NARMAX, time delay neural networks and recurrent neural networks are used for the same problem. The columns of the table specify the average prediction error of the 6 test sets for each controller. All methodologies used the same training and test data sets previously described.

The errors obtained for all the controllers using the SA approach hand in hand with the FIR methodology are much better than the ones obtained by the inductive methodologies presented in table 3. Moreover, the highest MSE_{test} of 4.76% obtained with the $1-Q$ cost function is half the value of the lower error obtained with these methodologies, i.e. 9.3%. Therefore, in this application, both cost functions can be considered good for the task at hand. The $1-Q$ cost function needs less time to be evaluated but the performance with respect to the test set prediction is lower. Contrarily, the MSE_{train} cost function is more expensive from the CPU time point of view but the performance is higher. The user should decide which cost function to use taking into account the size of the optimization problem and his/her own needs.

Table 4 contains the partition results of the other four CNS controllers. The random initial partition option has been used in all the executions. The SA algorithm has been executed 40 times for both cost functions for each controller. The final partition, the value of the cost function and the mean MSE of the 6 test data sets are presented for each controller and cost function. An $*$ means that that partition is the best possible one, and therefore it is the optimal solution. As can be seen in table 4 the optimal solution is reached for both cost functions in all partitions. The CPU time and number of generated solutions are equivalent to those of the CR controller in tables 1 and 2. It is interesting to analyze the MSE_{test} of the HR, PR, MC and Vt controllers. The errors of the test sets obtained when the MSE_{train} cost function is used are quite good for all controllers, and much better than the ones obtained using the inductive methodologies of table 3. However, this is not the case for all controllers when the $1-Q$ cost function is used. Notice that, although the SA algorithm finds both the best solution and good suboptimal solutions, the prediction errors of the test data sets obtained are of the same order of magnitude than the ones obtained by the NARMAX, time delay and recurrent neural networks, particularly for the MC and VT controllers. In this case, the quality measure used by the FIR methodology is not doing a good job. It can be interesting to study alternative quality measures for the task at hand.

4 Conclusions

In this paper the usefulness of a simulated annealing approach for the automated definition of fuzzy sets in the identification of human central nervous system FIR models has been shown. Two cost functions have been evaluated and compared from the perspective of their performance and computational time. The results

obtained in the CNS applications are much better than the ones obtained by other inductive methodologies such as NARMAX, time delay neural networks and recurrent neural networks.

Acknowledgements. The authors thank the valuable suggestions of F.E. Cellier of the University of Arizona, USA and P. Villar of the Universidad de Vigo, Spain. This research was supported by Spanish Consejo Interministerial de Ciencia y Tecnología (CICYT), under project DPI2002-03225.

References

1. F. Cellier. Continuous System Modeling. Springer-Verlag, New York 1991.
2. A. Nebot. Automatic tuning of fuzzy paryitions in inductive reasoning. In *Proceedings CIARP'03: 8th Iberoamerican Congress on Pattern Recognition*, Page 8, Havana, Cuba 26–29 November 2003. Under revision.
3. A. Nebot, F. Cellier, and M. Vallverdú. Mixed quantitative/qualitative modeling and Simulation of the cardiovascular system. *Computer Methods and Programs in Biomedicine* 55:127-155 1998.
4. C.D.G.J.S. Kirkpatrick and M. P. Vecchi. Optimization by simulated annealing. Science, 220:671–680, 1983.