Neurocomputing 72 (2009) 1656–1667

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Computational framework for behavioural modelling of neural subsystems

M.T. Signes, J.M. García, G. de Miguel *, H. Mora

Specialized Processor Architectures Laboratory, University of Alicante, Spain

ARTICLE INFO

Article history:
Received 14 December 2007
Received in revised form
2 May 2008
Accepted 4 September 2008
Communicated by T. Heskes
Available online 15 October 2008

PACS:
02.70 Wz
07.05 Bx
07.05 Tp
87.17 Uv
87.18 Hf

Keywords:
Behavioural modelling
Neural subsystems
Computer algebra
Hardware implementation

ABSTRACT

This paper presents a new approach to the problem of modelling living system dynamics. Our point of view claims the fact that behind the biological apparent complexity, a hidden simplicity may appear when a suitable modelling is developed. The framework is inspired on the computing features of biological systems by involving a set of elementary standard behaviours that can be combined in order to emulate more complex behaviours. The algebraic formalization is based on both a recursive primitive operation defined by a table which models the elementary behaviours and a multilevel operating mode that carries out behaviour combinations. A parametric architecture implements the model, providing a good trade-off between time delay calculation and memory requirements. In this paper, the simulation of neural subsystems is considered as an application. The comparison with other simulation techniques outlines the capabilities of our method to provide an accurate modelling together with a very simple circuit implementation.

1. Introduction

The ability of the computational technologies to approach the dynamics of the biological systems still remains as a big challenge. Artificial computation provides useful methods, techniques and computing models for contrasting, ordering and interpreting the huge amount of data collected from in vivo or in vitro experiments on living organisms. On one hand, biophysical modelling relates electric neuronal parameters such as voltages and currents by introducing differential equations [25,20,34] and has led to important advances in fundamental, cellular research on systems physiology [37,11,18,19,22]. On the other hand, mathematical modelling deals with the collective properties of the systems without explicitly relating them to the physical structure involved [14,21,26,1,23,8]. Enhancing the trend of the mathematical scope by adding some restrictions that must be fulfilled by the parts of the whole system, the complex system approach outlines the emergency of new properties that cannot be explained as the result of the combined individual features of the involved parts [36,3]. In some cases, biological process modelling can take the advantages stemming from both qualitative (behavioural) and quantitative (measures) information at the same time. Some recent approaches like genomics [7,5] proteomics [4,39] or metabolomics [9,12] deal with a huge quantity of information which has to be managed at different biological levels. The main point is how to find the simplicity hidden behind the biological complexity in order to achieve an accurate understanding of how the living world manages information, materials or energy [17]. Synthetic biology provides a mean to explore the biological landscape by designing and building artificial systems that behave as biological ones [6,10]. This approach provides a scientific and technological framework to support the functionality of biological system synthesis.

Our approach relates the apparent biological "complexity" to the mathematical operations and computational models used in order to explore the living world. If we take on board biological systems under the scope of the extended machine computational model, we have to consider that the simpler the primitive operations are (sum and multiplication), the more complex the function evaluation is (higher and higher computing levels are needed). So, a lower computing level can be achieved by introducing new primitives whenever they assume intrinsically part of the complexity. These issues are suitable for the case of living beings if we consider them as computing systems: the more

* Corresponding author.
E-mail addresses: teresa@dtic.ua.es (M.T. Signes), juanma@dtic.ua.es (J.M. García), demiguel@dtic.ua.es (G. de Miguel), hmora@dtic.ua.es (H. Mora).
URL: http://www.ua.es/i2rc/index2-eng.html (G. de Miguel).

0925-2312/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2008.09.008

sophisticated the formal tools are, the simpler will be the resulting model. In order to guide the formalization, as we deepen in the computing features of biological systems, some basic principles appear. The outputs of biological systems rely on both actual inputs and past inputs, as if they had "memory" [2]. Furthermore, the outputs are not too sensitive to the variations of the inputs in order to allow error tolerance and graceful degradation [28]. Thus, their behaviours are stable inside large intervals of their parameters values. Our proposal has been inspired by these premises. The proposal presents a framework for modelling the behaviour of biological systems without linking it to the particular structure under study. The framework involves a set of elementary standard behaviours that can be combined so that to emulate more complex ones. The algebraic formalization is based on both a recursive weighted primitive defined by a table which models the elementary behaviours and a multilevel operating mode which carries out the behaviour combination. The recursive approach supports the calculation of new values using previous ones, as if the system could "remember", whereas the primitive parameter values determine the stability intervals for different elementary behaviours. A very simple circuit implements this approach with a satisfying trade-off between area and time delay. In this context, this approach has been successfully tested to emulate neuron spiking.

The paper is structured in five parts. Following the introduction, Section 2 presents the main features of the computational framework. The primitive level operation and the recursive operation mode are defined as well as the computation at upper levels. In Section 3, an implementation based on look-up tables (LUTs) is discussed and an estimation of the time delay calculation and memory requirements is provided. Section 4 develops some applications which deal with the simulation of neural structures. The case of the epileptiform bursts in the CA3 region of the hippocampus is presented. It outlines that the biophysical modelling can be improved by the simulation based on our model. Additionally, the simulation of the central pattern generator (CPG) which controls the swim rhythm of the mollusc *Tritonia diomedea* is also successfully compared with the simulation provided by our method. Finally, Section 5 summarizes the concluding remarks of this research.

2. Computational framework

This section presents the computational framework. It involves the definition of the primitive operation which generates the elementary behaviours as well as the multilevel operating mode which provides more complex behaviours.

2.1. Primitive operation level

The primitive operation has been defined by a weighted sum denoted as $\otimes$ (see Eq. (1))

$$\otimes : \mathbb{R} \times \mathbb{R} \to \mathbb{R},$$

$$(a, b) \to a \otimes b = \alpha a + \beta b,$$

$$(\alpha, \beta) \in \mathbb{R}^2. \tag{1}$$

This formula can be considered as a primitive operation, just as the usual computational primitives addition and shift. The generic definition of the new primitive can be achieved by a two-dimensional table in which the cells store combinations of the weighting parameters. The table in Fig. 1 defines the operation for the particular case of integer values, for binary sign-magnitude representation and for $k = 1$ (k stands for the number of significant bits in the representation). The arguments have been represented in binary and decimal notation and the results are referred in a generic way, as combinations of the parameters α and β.

The same operation can be represented for greater values of k (see table in Fig. 2 for $k = 2$). The central cells are equivalent to those of the table shown in Fig. 1. The number of cells in a table is $(2^{(k+1)} - 1)^2$ and it only depends on k. The cells are organized as concentric rings centred in 0. It can be noticed that increasing the value k causes a growth in the table which adds more peripheral rings. The number of rings increases at a rate of 2^k when k increases one unit. The smallest table is defined for $k = 1$, but the same information about the operation $\otimes$ is provided for any value of k.

The operation $\otimes$ is performed when the arguments (a, b) address the table and the result is picked up from the corresponding cell. The first argument (a) addresses the row whereas the second (b) addresses the column. When the precision of the arguments n is greater than k, these must be fragmented in k-sized fragments in order to perform the operation. So, t double accesses are necessary to complete t cycles of a single operation (remember $n = k \cdot t$). A single operation requires picking up from a table so many partial results as fragments are contained in the argument. The overall result is obtained by adding t partial results, according to their position. It can be noticed that any pair (α, β) defines a new table, that is to say, a new instantiation of the primitive.

a ⊗ b	01 = 1	10=00 = 0	11 = -1
01 = 1	$\alpha+\beta$	α	$\alpha-\beta$
10 = 00 = 0	β	**0**	$-\beta$
11 = -1	$-\alpha+\beta$	$-\alpha$	$-\alpha-\beta$

Fig. 1. Definition of the operation $\otimes$ for $k = 1$.

a ⊗ b	011= 3	010= 2	001= 1	100=000=0	101= -1	110=-2	111=-3
011= 3	$3\alpha+3\beta$	$3\alpha+2\beta$	$3\alpha+\beta$	3α	$3\alpha-\beta$	$3\alpha-2\beta$	$3\alpha-3\beta$
010= 2	$2\alpha+3\beta$	$2\alpha+2\beta$	$2\alpha+\beta$	2α	$2\alpha-\beta$	$2\alpha-2\beta$	$2\alpha-3\beta$
001= 1	$\alpha+3\beta$	$\alpha+2\beta$	$\alpha+\beta$	α	$\alpha-\beta$	$\alpha-2\beta$	$\alpha-3\beta$
100=000= 0	3β	2β	β	**0**	$-\beta$	-2β	-3β
101= -1	$-\alpha+3\beta$	$-\alpha+2\beta$	$-\alpha+\beta$	$-\alpha$	$-\alpha-\beta$	$-\alpha-2\beta$	$-\alpha-3\beta$
110= -2	$-2\alpha+3\beta$	$-2\alpha+2\beta$	$-2\alpha+\beta$	-2α	$-2\alpha-\beta$	$-2\alpha-2\beta$	$-2\alpha-3\beta$
111= -3	$-3\alpha+3\beta$	$-3\alpha+2\beta$	$-3\alpha+\beta$	-3α	$-3\alpha-\beta$	$-3\alpha-2\beta$	$-3\alpha-3\beta$

Fig. 2. Definition of the operation $\otimes$ for $k = 2$.