PULSED PARA-NEURAL NETWORK (PPNN) SYNTHESIS IN A 3-D CELLULAR AUTOMATA SPACE

Andrzej Buller[1]*, Hendrik Eeckhaut*[2] *& Michal Joachimczak*[3]

[1]ATR Human Information Science Laboratories
2-2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0288 Japan
buller@atr.co.jp

[2]Ghent University, Department of Electronics and Information Systems
Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium
Hendrik.Eeckhaut@elis.rug.ac.be

[3]Gdansk Artificial Brain Research Initiative
ul. Grunwaldzka 92/98 m.56, 80-244 Gdansk, Poland
guhru@klub.chip.pl

ABSTRACT

This paper deals with a synthesis of Pulsed Para-Neural Networks (PPNN) in a 3-D Cellular Automata space. In its essence, PPNN is a set of simple processing units that change their states only in certain discrete moments of time denoted t, t+1, t+2, ... The inlets to and outlets from the units are located in such a way that a given unit may (but does not have to) send a pulse to and only to its nearest neighbors. There are three kinds of processing units: red cells, yellow cells, and blue cells. A red cell emits a pulse at t+2 if and only if the value of its counter at t+1 gets equal to or greater than 2, where the state of the counter for t+1 is the state for t plus the weighted sum of pulses incoming in t. Each weight is associated with one and only one inlet to the cell and may be equal to 1, 0 or –1. Every red cell zeroes its counter when emitting a pulse. Every yellow cell emits a pulse in t+1 if and only if one and only one pulse entered it in t. Every blue cell emits a pulse in t+1 if a pulse entered it in t through its only inlet. A red cell works in the way approximating the behavior of a neural soma which cumulates excitation/inhibition in both time and space. A yellow cell represents a formal neuron whose inputs provide a pre-synaptic inhibition to all other inputs. Since a blue cell is a 1-clock delay, a string of adjacent blue cells may model an axon.

Despite the simplicity of the paradigm of computation of PPNN, a number of useful devices has been created in its framework. We present (1) an associative memory to be filed via reinforcement learning (what is remembered is a set of phases of pulses circulating in closed loops made of cells), and (2) a spiking neuron that non-linearly cumulates excitation and responds with a changeable frequency of produced pulses, (3) an adjustable timer and (4) an object location recognizer. We also present tools and methods used for PPNN synthesis.

1. INTRODUCTION

PPNN is a set of simple processing units that change their states only in certain discrete moments of time denoted t, t+1, t+2, ... The inlets to and outlets from the units are located in such a way that a given unit may (but does not have to) send a pulse to and only to its nearest neighbors.

The idea of using a single cell in a Celular Automata as a model of a neuron body, and cell trains as collaterals, appeared during the research for a hardware platform for ATR's Artificial Brain project, and was named CoDi ("Collect and Distribute"). First version of CoDi-style PPNN with a mechanism of its evolutionary synthesis was written in C++ and run on SUN workstation by Felix Gers [4].

CoDi-style PPNN uses three kinds of processing units. In order to facilitate its narrative description, let us call them red cells, yellow cells, and blue cells. A red cell emits a pulse in t+2 if and only if the value of its counter in t+1 gets equal to or greater than 2, while the state of the counter for t+1 is the state for t plus weighed sum of pulses incoming in t. Each weight is associated with one and only one inlet to the cell and may be equal to 1, 0 or –1. Every red cell zeroes its counter when emitting a pulse. Every yellow cell emits a pulse in t+1 if and only if one and only one pulse entered it in t. Every blue cell emits a pulse in t+1 if a pulse entered it in t through its only inlet. A red cell works in the way approximating the behavior of a neural soma which cumulates excitation/inhibition in both time and space. A yellow cell represents a formal neuron whose inputs provide a pre-synaptic inhibition to all other inputs.

Since a blue cell is a 1-clock delay, a string of adjacent blue cells may model an axon.

ATR's Brain Building Group successfully evolved CoDi-based timers and a switchable dual function module, while the members of Gdansk Artificial Brain Research Initiative (GABRI) successfully evolved CoDi-based frequency-to-delay converter and erasable memory [2], as well as a spike-train memorizer [1].

Although a dedicated FPGA-based hardware, called CBM (CAM-Brain Machine), for evolution and simulation of CoDi-based neural networks has been constructed [4], one still has not managed to evolve more sophisticated devices. Hence, while theoretical investigations and research towards more efficient genetic algorithm for PPNN are being continued, alternative methods of PPNN synthesis are also being developed at ATR Human Information Science Labs, Kyoto, Japan, Gdansk Artificial Brain Research Initiative (GABRI), Poland, as well as at the Department of Electronics and Information Systems, Ghent University, Belgium. Using tools and methods developed in these centers a number of complex devices for artificial brains has been created.

In the next sections we provide a formal model of PPNN, a description of tools and methods currently used for PPNN synthesis, as well as selected testing applications.

2. PPNN MODEL

Let us start from some basic notions:

$\mathbf{B} = \{0, 1\}$;

$\mathbf{F} = \{-1, 0, 1\}$;

$\mathbf{0} = (0, 0, 0)$;

$\mathbf{V} = \{ (v_x, v_y, v_z) \in \mathbf{F}^3 \mid v_x^2 + v_y^2 + v_z^2 = 1 \}$;

$\mathbf{T}$, $\mathbf{Z}$ are spaces of integers ($\mathbf{T}$ refers to time);

β is a one-argument function that for a given statement returns 1 when the statement is true, while otherwise it returns 0, e.g. $\beta(1=2) = 0$;

$\underline{\ }$ is a monadic operator that when applied to 0 gives 1, while when applied to 1 it gives 0, i.e. $\underline{0} = 1$, $\underline{1} = 0$.

Based on the above nomenclature PPNN has been defined.

Definition 1 (based on [1]).

A quadruple $\langle \mathbf{N}, w, x, c \rangle$ is PPNN if:

$\mathbf{N} \subset \mathbf{Z}^3$,

$w : \mathbf{N} \times \mathbf{V} \cup \mathbf{0} \rightarrow \mathbf{F}$,

$\mathbf{N}' = \{q \in \mathbf{Z}^3\text{-}\mathbf{N} \mid \exists_{p \in \mathbf{N},\, v \in \mathbf{V}}\ w_{p,v} \neq 0,\ p+v = q\} \neq \varnothing$,

$x : \mathbf{N} \cup \mathbf{N}' \times \mathbf{T} \rightarrow \mathbf{B}$,

$c : \mathbf{N} \times \mathbf{T} \rightarrow \mathbf{Z}$,

$\forall_{p \in \mathbf{N},\, t \in \mathbf{T}}$

$x_{p,t+1} = w_{p,\mathbf{0}}\beta(c_{p,t} \geq 2) + \underline{w}_{p,\mathbf{0}}\beta(\Sigma_{v \in \mathbf{V}}\ w_{p,v}x_{p+v,t}=1)$

$c_{p,t+1} = \underline{x}_{p,t+1}c_{p,t} + \Sigma_{v \in \mathbf{V}}\ w_{p,v}x_{p+v,t}$.

Set $\mathbf{N}'$ contains input points to $\langle \mathbf{N}, w, x, c \rangle$. Any $\mathbf{N}'' \subseteq \mathbf{N}$ can be a set of output points.

3. PPNN SYNTHESIS

Two separate problems must be solved in order to obtain a desired PPNN: logic synthesis and routing. As for the logic synthesis, a finite state machine must be defined. The routing must be done within a 3-D discrete space. Genetic algorithm that adequately can solve these problems have not been found yet. Hence, less fashionable methods of PPNN synthesis are being developed, starting from computer-aided handcrafting.

One approach, developed by Hendrik Eeckhaut, consists of interactive hierarchical manipulation of cells in the visualized 3-D space [3]. Another approach, implemented by Michal Joachimczak as a software tool called *NeuroMaze Works* (*NMW*), provides a PPNN-designer worksheets for visualizing and editing layers of the Cellular Automaton's space in which he can graphically define values of $w_{p,v}$ for all visible p. When $w_{p,\mathbf{0}}=1$, p is visible as a red square. When $w_{p,\mathbf{0}}=0$ and $w_{p,v}=1$for one and only one $v \in \mathbf{V}$, p is visible as a blue square. When $w_{p,\mathbf{0}}=0$ and $w_{p,v}=1$for two or more values of $v \in \mathbf{V}$, p is visible as a yellow square. Non-zero values of $w_{p,v}$ for $v \neq \mathbf{0}$ are visualized as small colored triangles. At any stage of designing, one can activate selected cells and observe how the activation propagates. Designs can be displayed as 3-D pictures that can be zoomed, rotated, and provided with spike-trains to be processed. A Cellular Automaton space for a single design consists of 24×24×24 cells. Under *NMW* thousands of such designs can be easily interconnected towards very-large-scale PPNNs.

Future versions of *NMW* are intended to combine the worksheet-based approach to PPNN handcrafting with direct manipulation on cells in visualized 3-D space, as well as to include several tools for semi-automated creation of standard PPNN sub-structures, for example an automated creation of an axonic path between defined cells.

4. PPNN APPLIED

Despite the simplicity of PPNN's paradigm of computation, a number of useful devices has been created in its framework. We present (1) an associative memory to be filed via reinforcement learning (what is remembered is a set of phases of pulses circulating in closed loops made of cells), (2) a spiking neuron that non-linearly cumulates excitation and responds with a

changeable frequency of produced pulses, (3) an adjustable timer and (4) an object location recognizer.

The CBM we use for our experiments provides a cellular automata working space consisting of 64,640 24×24×24-cell modules. Although a number of experiments with multimodule systems has been performed, in this paper we present only some single-module applications.

4.1. Associative Memory

The Associative Memory that has been successfully built using NeuroMaze technique is a PPNN-based device that learns to map a set of six possible stimuli onto a set of eight possible reactions [1]. Every reaction is represented as a spike train of length 8 containing only a single 1. The position of the 1 identifies a particular reaction. A given stimulus is represented as a single spike provided to one of six data inputs to the device. After every stimulus a user is expected to see reaction and then either reward or punish the device. There are two learning inputs, one for a rewarding spike and one for a punishing spike. If a rewarding spike is provided, the device will react in this way for succeeding occurrences of the same stimulus . If a punishing spike is provided, the device will react differently at the next occurrence of this stimulus..

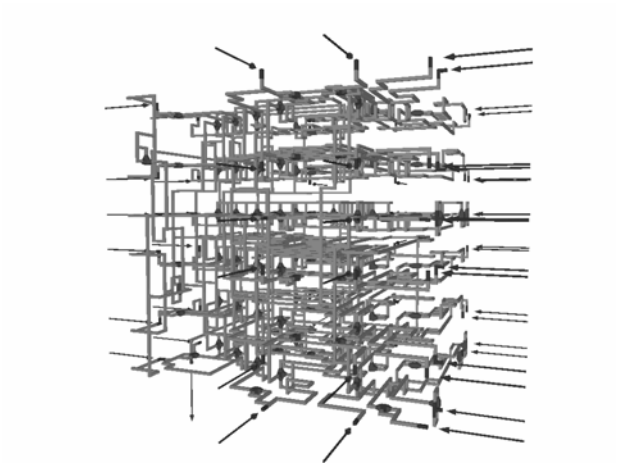

Fig. 1. Associative Memory based on a Pulsed Para-Neural Network

4.2. Spiking Neuron

The spiking neuron (SN) that has been successfully built using NeuroMaze technique is a PPNN-based device that non-linearly cumulates excitation and responds with a changeable frequency of produced pulses.
According to Buller et al. [1]:

$$f_{t+\Delta+\delta} \approx \sigma_{1,t+\delta}\, \beta(\forall_{\tau\in[t;\, t+8]} s_{1,\tau}=1)/16 + \sigma_{2,t+\delta}\, \beta(\forall_{\tau\in[t;\, t+8]} s_{2,\tau}=1)/16$$

where $f_{t+\Delta}$ is the mean frequency of 1s produced by SN in time $t+\Delta$. $s_{1,t}$, $s_{2,t}$ are the main inputs of SN, $\Delta+\delta$ is the pulse propagation time (from input to output), while $\sigma_{1,t}$, $\sigma_{2,t}$ are adjustable weights changing according to the formula:

$$\sigma_{i,t+\delta} = \_\lceil(\ \sigma_{i,t} + u_{i,t} - h_{i,t}\ )$$

where $u_{i,t}$ is the input for a pulse increasing i-th weight, $h_{i,t}$ is the input for a pulse decreasing i-th weight, while $\_\lceil : \mathbf{Z} \rightarrow \mathbf{Z}$ is a function that works according to the formula: $\_\lceil(n) = 0.5(|n| - |n-8|) + 4$.

4.3. Adjustable Timer

Another useful module is the adjustable timer. This device can block or let pass spike trains for selectable periods of time. There are multiple inputs, which are related to different time intervals. Depending on the input on which a triggering pulse arrives, this device will temporarily stop blocking another spike train for a preprogrammed period of time. A module with periods 10, 20, 30, 40, 50, 60, 70 and 80 was built, but the technique is more generally useful. All time periods are possible. For each time period one wants, one only needs an input, 17 cells, the same number of cells as the length of your period plus some additional cells for interconnection (a complete module has 188 inputs and 13824 cells).
If a spike arrives to unlock the timer for a certain period, while the device lets a spike train pass for another period (via another input), the duration of the period will be that of the longest of both.

4.4. Object Location Recognizer

A simple PPNN named OLR (Object Location Recognizer) has been synthesized and loaded into the CBM (a dedicated hardware for PPNN-based systems) to control a robot consisting of 5-DOF arm with a gripper and mounted camera (Fig. 2). The image produced by the camera is divided into 8 × 8 rectangles. 4 squares in the center of the image form a "blind spot" (BS). There are 4 zones: on-the-left-of-BS zone, on-the-right-of-BS zone, beneath-BS zone and above-BS zone. Each zone consists of 24 squares, shares 9 squares with one of three remaining zones and other 9 squares with another of the three remaining zones. Regardless the overlapping of some squares, each zone is represented by 24 inputs to OLR. When a certain threshold number of pixels in a given square becomes blue, related ORL's input gets a spike, which results in the production of a spike in one or two of ORL's outputs. Owing to this the robot's arm "knows" how to move to grab a blue object.

Fig. 2. Robotic arm controlled using a PPNN loaded into a dedicated hardware (CBM).

6. CONCLUDING REMARKS

Despite the simplicity of PPNN's paradigm of computation, a number of useful devices can be created in its framework. We presented (1) an associative memory to be filled via reinforcement learning (what is remembered is a set of phases of pulses circulating in closed loops made of cells), (2) a spiking neuron that non-linearly cumulates excitation and responds with a changeable frequency of produced pulses, (3) an adjustable timer, and (4) an object location recognizer. The devices have been created using tools for computer-aided PPNN handcrafting we develop. We also use FPGA-based dedicated hardware for our experiments. This provides us with a cellular automata working space consisting of 64,640 24×24×24-cell modules. The ultimate version of the hardware may be a set of stand-alone modules of microscopic size. Armed with such supporting software and hardware, PPNN paradigm seems to be very useful for building brains for advanced robots.

Acknowledgements. This research was conducted as a part of the *Research on Human Communication* supported by the Telecommunications Advancement Organization of Japan.

The research in Ghent was made possible because Tibotec-Virco NV placed a CBM at our disposal; this support is gratefully acknowledged.

REFERENCES

[1] A. Buller, M. Joachimczak & J. Bialowas, "NeuroMaze: A New Method of Pulsed Neural Network Synthesis", *Proc. Obf The 7th Int. Symposium on Artificial Life anad Robotics (AROB 7th '02), Beppu, Oita, Japan*, pp. 648-64916-18 January 2002.

[2] H. de Garis, A. Buller, M. Korkin, F. Gers, N.E. Nawa & M. Hough, "ATR's Artificial Brain ("CAM-Brain") Project: A Sample of What Individual "CoDi-1Bit" Model Evolved Neural Net Modules Can Do with Digital and Analog I/O", *Proceedings of The First NASA / DoD Workshop on Evolvable Hardware, Pasadena, California*, pp. 102-110. July 19-21, 1999.

[3] H. Eeckhaut, „A graphical programming environment for the CAM-Brain Machine"(in Dutch). *Master's thesis, University Ghent (ELIS), Ghent*, 2001.

[4] F. Gers, H. de Garis & M. Korkin, „CoDi-1Bit: A Simplified Cellular Automata based Neuron Model", *Evolution Artificiele 97, Nimes, France*, pp. 211-229, 22 October, 1997.

[5] M. Korkin, H. de Garis, N.E. Nawa & W.D. Rieken, „ATR's CAM-Brain Project: CAM-Brain Machine (CBM) and Robot Kitten (Robokoneko) Issues", In: A. Drobnikar et al. (Eds.) *Artificial neural Nets and Genetic Algorithms*, Springer: Wien, pp. 107-110, 1999.