ATR’s CAM-Brain Project:
The Evolution of Large-Scale Recurrent Neural Network Modules

Norberto Eiji Nawa
Bio-Electronics Lab. Graduate School of Engineering,
Nagoya University and Evolutionary Systems Dept.,
ATR - Human Information Processing Research Laboratories,
2-2 Seika-cho, Kyoto 619-0288, Japan
xnawa@hip.atr.co.jp, http://www.hip.atr.co.jp/~xnawa

Michael Korkin
Genobyte, Inc., 1319 Spruce Street, Suite 210,
Boulder CO 80302, USA
korkin@genobyte.com, http://www.genobyte.com

Hugo de Garis
Evolutionary Systems Dept., ATR - Human Information Processing Research Laboratories,
2-2 Seika-cho, Kyoto 619-0288, Japan
degaris@hip.atr.co.jp, http://www.hip.atr.co.jp/~degaris

Abstract This paper describes ongoing ATR’s CAM-Brain Project, which is an attempt to build large-scale neural networks ('artificial brains') in a special hardware called "CAM-Brain Machine" (CBM). At the time of writing (March 1998), the project is making efforts on two fronts - the construction of the CBM, that is scheduled to be operational in the summer of 1998, and attempting to find an efficient and effective representation for the binary signaling of ATR’s CAM-Brain Machine (CBM), using the so-called "CoDi-1Bit" model. The CBM is an FPGA based hardware accelerator which updates 3D cellular automata (CA) cells at the rate of 100 billion a second, allowing a complete run of a genetic algorithm with tens of thousands of CA based neural net circuit growths and hardware compiled fitness evaluations. It is hoped that by using such a device, it will become practical to evolve 10,000s of neural net modules and then assemble them into humanly defined RAM based artificial brain architectures which can be run by the CBM in real time to control robots, e.g. a robot kitten. Before large numbers of modules can be assembled together, it is essential that the individual modules have a good functionality and evolvability. The "CoDi-1Bit" CA based neural net model uses 1 bit binary signaling, so a representation needs to be chosen based on this fact. This paper discusses the merits and demerits of a representation that we call "Spike Interval Information Coding" (SIIC) and presents some simulation results obtained so far concerning the evolution of simple functional modules and the performance of the SIIC representation.

1In the Proceedings of the 1998 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'98), July 13-16, 1998, Las Vegas, Nevada, USA

Keywords: Artificial Brain, CAM-Brain, Artificial Neural Networks, Spike Interval Coding
1 Introduction

The CAM-Brain Project at ATR Labs aims to construct a large-scale brain-like neural network system. If the project succeeds and our expectations are fulfilled, these "artificial brains" will have a large number of potential applications in several different fields, from 'smart' domestic appliances to speech processing and robot control. Of course, up to now, this is pure speculation, and we admit there is still a long way to go before we can talk in more concrete terms. However, we believe that to realize a system that possesses a level of functionality and structural complexity similar to real biological brains, the most appropriate way, if not the only way, is to evolve them, as happened in nature.

The fundamental approach of the CAM-Brain Project is the growth/evolution of large-scale neural networks. Since the dawn of the Project [1], Cellular Automata (CA) have been chosen as the medium in which to grow the neural networks. CAs meet the requirements of generality and especially scalability, necessary for simulating large-scale systems. Moreover, the parallel nature of CAs allows their transposition into hardware, where higher speeds can be achieved. The CA based neural model initially used [2] suffered from an explosion of states and transition rules that blocked any attempt to implement it in hardware. Due to this problem, a new model called "CoDi" (from COlect and DIstribute) was proposed [3], greatly simplifying the system and for the first time allowing the implementation of the system in special hardware, namely XC6264 FPGAs (described below) which update the CA space at a rate of 100 billion cells a second, and should be able to perform a complete run of a genetic algorithm (with 10,000s of circuit growths and evaluations) in about a second.

Advances in hardware technology led to the development of devices called Field Programmable Gate-Arrays (FPGAs). FPGAs are hardware devices that can be reconfigured at run-time to perform different logic functions, wedging the flexibility of software with the speeds of hardware. This motivated the design/construction of a specific computer, called the CAM-Brain Machine (CBM) [4], for the evolution of neural networks under the CoDi model. The CBM will grow 16,000 neural network modules of roughly 10,000 3D CA cells each, updating 100 billion cells/second.

The experiments reported in this paper aim to tackle a fundamental issue of the project: how to interpret the signals that enter and leave the neural network modules? The CoDi-1Bit model evokes neural networks whose signals which traverse the connections are digital, i.e. binary 0's and 1's. The question is, what kind of representation schemes should be used in order to extract useful information from the input and output signals of the neural network modules. This paper uses a "Spike Interval Information Coding" (SIIC) representation, based on results from the field of neuroscience reported in [5], which deals with the theory of information encoding in neural systems. The results described in this paper are encouraging so far, and indicate that the SIIC is a suitable representation scheme for the CoDi model.

2 The CoDi-1Bit Cellular Automata Based Neural Net Model

The model is called "CoDi" due to the "COlect and DIstribute" nature of its neural signals. Its goal was to make neural network functioning much simpler compared to the older CAM-Brain model developed in 1993 and 1994 [6, 1], so as to be able to implement the model directly in electronics and thus to evolve neural net modules at electronic speeds.

In order to evolve one neural network module, a population of modules is run through a genetic algorithm for several hundred generations. (The genetic algorithm currently used is a canonical one, with ordinary selection, crossover and mutation operators.) Each module evaluation consists of growing a new set of axonic and dendritic trees which interconnect
the neurons in the 3D cellular automata space, then running the module to evaluate its performance (fitness).

The neural networks constructed by the CoDi model resemble neural networks of the parallel distributed processing paradigm [7]. The nodes are not necessarily organized into layers. Connections between nodes do not necessarily have the same length, which implies that signals take different times to traverse the connections. Connections have no weights and the incoming signals are binary. Moreover, the number of neurons in a neural network module is of the order of hundreds, considerably higher than the number found in common neural networks. There is no back-propagation like training. The networks are evolved to satisfy a given evaluation criterion. Recurrency is free between the nodes, and the structure is defined by the evolutionary process.

The CoDi model cited above [3] operates as a 3D cellular automata. Each cell is a cube which has six neighboring cells, one for each of its faces. By loading a different phenotype code into a cell, it can be reconfigured as a neuron, an axon, or a dendrite. Neurons are configurable on a coarser grid, namely one per block of 2*2*3 CA cells. In a neuron cell, five (of its six) connections are dendritic inputs, and one is an axonic output. An accumulator sums incoming signals and fires an output signal when a threshold is exceeded. Each of the inputs can perform an inhibitory or an excitatory function (depending on the neurons chromosome) and either adds to or subtracts from the accumulator value. The neuron cell’s output (axon) can be oriented in 6 different ways in the 3D space.

A dendrite cell also has maximum five inputs and one output, to collect signals from other cells. The incoming signals are passed to the output according to a given function. For instance, if an XOR function is used, the output is active when only a single input is active. Two or more active inputs block each other. The XOR dendrite is more plausible from the biological point of view, since a similar phenomenon occurs in real dendrites in animals. An axon cell is the opposite of a dendrite. It has 1 input and maximum 5 outputs, and distributes signals to its neighbors. Before the growth begins, the module space consists of blank cells, which are used to grow new sets of dendritic and axonic trees during the growth phase. Blank cells perform no function in an evolved neural network.

As the growth starts, each neuron continuously sends growth signals to the surrounding blank cells, alternating between “grow dendrite” (sent to the neuron’s dendritic connections) and “grow axon” (sent to the axonic connection). A blank cell which receives a growth signal becomes a dendrite cell, or an axon cell, and further propagates the growth signal, being continuously sent by a neuron, to other blank cells. The direction of the propagation is guided by the growth instructions attached to the cell. These local instructions indicate the directions that the growth signal should be propagated to and consists of a bit for each face of the cube cell. The growth signal is propagated to those directions whose corresponding bit is set to 1 (except for the direction where that signal comes from).

This mechanism allows the growth of a complex 3D system of branching dendritic and axonic trees, with each tree having one neuron cell associated with it. The trees can conduct signals between the neurons to perform complex spatio-temporal functions. The end-product of the growth phase is a phenotype bitstring which encodes the type and spatial orientation of each cell.

3 CAM-Brain Machine (CBM)

This section briefly describes the hardware implementation of the above CoDi-18Bit model, allowing CoDi neural net modules to be grown in hardware.

The CAM-Brain Machine (CBM) was especially designed to support the growth and signaling of neural networks built by the CoDi model. The CBM should fulfill the need for high speed, when simulating large-scale binary
neural networks, a necessary condition when one is concerned with performing real-time control. The hardware core is implemented in XC6264 FPGA chips, in which the neural networks will actually grow. A host machine will provide the necessary interface to interact with the hardware core. It is planned that the CBM will be used to grow 16,000 neural networks modules, each with approximately 10,000 cells. The modules will be organized in architectures defined in advance, so several neural network modules will be interconnected to form a functional unity. This machine should be built by the summer of 1998. For a complete description of the CBM, refer to [4].

4 The Spike Interval Information Coding (SIIC) Representation

The SIIC representation is inspired by the ideas presented in the book "Spikes: exploring the neural code", by Rieke et al. [5]. The book presents a novel hypothesis to explain how sensory signals are encoded in the action potentials or spikes that traverse neural systems. The classical theory of information encoded in neural signals was initiated by the work of E.D. Adrian, which strongly influenced the neuroscience community in the following years. Adrian’s basic idea was that information about the intensity of the stimulus is encoded in the rate of spikes it generates. The rate of spikes can be calculated by counting the number of spikes in a fixed time window, following the beginning of the stimulus.

However, the work of Rieke et al. provides a new explanation. It claims that, if the classical theory is correct, the information rates would be inefficient. Moreover, it observes that "(...) single neurons can transmit large amounts of information, on the order of several bits per spike." So the information should be contained not in the rates, which is a kind of averaging, but in the spikes trains themselves. The book [5] develops the theory and mathematical background to support its claims. Since this theory is beyond the scope of this paper, it will not be developed any further. Here, we simply take part of it, namely, given a train of spikes, how should one decode it, in order to obtain useful information.

The procedure presented in the book has a different motivation from the one used in this paper. In the book, the aim is to find an appropriate method to construct an estimation of the analog stimulus signal from the spike train. Whereas, at the current stage of our research, the motivation is to find a method to extract information from the 0-1 bits that are output from the neural network modules evolved in the CBM. Considering that a 0 represents the absence of a spike, and a 1 represents the presence of a spike, the problem is quite similar. In earlier experiments, we found that the CoDi model evolved well for the cases of single fixed position outputs. However, we were unable to achieve satisfactory results nor a suitable representation method when using multiple non fixed position outputs [8] (the so-called "unary" representation, where if N output surface neurons were firing at a given moment, the number N was being represented.) The spikes approach seems to be more suitable for the CoDi model in this sense, since it works with single fixed position outputs.

4.1 The SIIC Decoding Method

The procedure for decoding a spike train (the sequence of 0-1 bits), named SIIC, consists of convoluting it with a special "convolution filter" (Fig.1. The result obtained is called the estimated signal, which is a time-dependent signal that is output from the neural network module to be evaluated in a fitness calculation or to be used in another process. The convolution process is discrete, since the convolution filter, the spikes trains and the results are all discrete. The estimated signal is a digital representation of an analog signal, sampled at discrete time points, corresponding to the clock ticks. The SIIC decoding process is as follows:

1. Collect $m$ bits from the output of the neural network module.
2. The estimated signal must have size \( n \leq m \). Every point of the estimated signal is mapped to a point of the stream of bits collected in the previous step.

3. The filter and the bit stream are overlapped. The first point of the filter corresponds to the first bit of the stream and the same for the subsequent points.

4. To calculate the first point of the estimated signal, convolve the filter and the output stream, i.e. sum the values of the points of the filter where the corresponding point in the output stream is a 1.

5. The obtained value corresponds to the first value of the estimated signal. Then shift the filter to the next bit, so the first point of the filter corresponds to the second bit of the output stream, and repeat the procedure described in the previous step.

6. Repeat the procedure described in the two previous steps to calculate all the points of the estimated signal.

---

4.2 Experiments with CoDi and SIIC

The next step was to determine the evolvability of the CoDi model, i.e. would the CoDi model be able to generate neural network modules whose binary outputs, decoded by the SIIC method, satisfy a given fitness function? Two experiments were performed. The first task was the generation of time dependent signals, sinusoidal waveforms, and the second task, a simple evolution of a functional module.

4.2.1 Sine Waves

In this experiment, we attempted to evolve several periods of a sine wave (Figures 2, 3, 4, 5). The solid lines are the target sine waves, and the dashed lines are those obtained after 600 generations. The output stream collected from the neural network modules had length equal to 300 bits and were collected 30 tick clocks after the simulation started. The length of the estimated signal was 120 bits. A simple GA was used in a population of 30 chromosomes, and the CA module size was \( 24 \times 24 \times 18 \). 48 input points (neurons) were chosen from one of the faces of the cubic CA space and were constantly firing in order to bring a high level of input activity to the module. The output signal was collected from a point on the opposite face of the inputs. The estimated signal was normalized, so that its discrete time points would have values of the same order as a unitary sine wave.
Overall, the results obtained in the experiments described above show that, at least for the case of evolving sinusoidal waveforms, the SIIC method can achieve better results than the more simplistic schemes tried before [8] (e.g., unary representation, incremental up/down counter representation, Gray code representation). The results indicate the ability of the CoDi-SIIC scheme to evolve neural network modules that can output time dependent signals at least this complex.

4.2.2 Evolution of functional neural networks modules

In the previous experiments, all the neural network modules evolved had several inputs in one of the faces that were constantly firing. This was done to bring a high level of activity to the modules, i.e. potentially the output is influenced by a larger number of neurons in the CA space, and this can increase the potential of the module as a whole. However, the inputs were constant, so no input-output relation was developed. Functional modules will be necessary in any application, so in these experiments we tried to evolve neural network modules that can perform a given function ability as parameterized by their inputs.

Firstly, neural network modules that generate one, two, three and four periods of a sinusoidal waveform using the SIIC representation were evolved, as described before. When a fair level of accuracy was achieved, we collected the streams of spikes that generate the sinusoidal waveforms. These streams could be used as inputs for the functional modules. The functional modules have only a single input, not several, as in the previous experiments.

Then, the functional neural networks were evolved. The input stream which when convoluted gave the four period sinusoidal wave was input to a fixed position single neuron. The objective was to generate in the output a similar sinusoidal waveform of the same frequency but with a lead in phase of π/4. The modules were evolved for 40 generations and the best module was saved. The waveform obtained is shown Figure 6. The output follows the desired form. Then, a though test was performed. Using the same neural network module used to generate the previous figure, the input was changed. Instead of the stream of spikes that generates the four-period sinusoidal, a three period sinusoidal was input. The obtained output is shown in Fig.7. There is some degradation, which was expected, since in the evolution phase, the modules had no contact with this input. But it is interesting to notice that clearly 3 peaks are present in the output, which
is an indication that the evolved neural module is somewhat relating the output and input and that the degradation in the performance can be graceful if there is a high level of similarity between the inputs. In other words, the CoDi modules are capable of generalizing. The modules have 24*24*18 cells, but the input is fed to only a single cell. Since no other source of activity is present, what comes from the output is the result of processing the single input. Figure 8 and 9 shows the waveforms when two and one period of sinusoidal waves in spikes form are input in the evolved module. The output waveforms have little resemblance with the desired ones, which is expected.

5 Conclusions

This paper described a "Spike Interval Information Coding" (SIIC) representation to decode the binary outputs of the neural network modules built by the CoDi model and to be run in the CAM-Brain Machine. The results obtained indicate that the SIIC method and the CoDi model can evolve time dependent waveforms, as simple as sinusoids, with fair levels of accuracy. The decoding process uses a filter, that is convoluted with a spike train to generate the estimated signal. In the experiments described in this paper, the filter was defined in advance and remained unchanged through the evolutionary process. Since the shape of the filter greatly influences the evolvability of the system, we feel that it too could be evolved. Moreover, further clarifications concerning the decoding method itself are necessary.

In the second part of the experiments, rudimental functional modules were evolved. The results show that the evolved neural network modules perform some processing on the single inputs that result in the waveforms at the output. Also, for inputs which are similar to those used in the evolution phase, the modules can generalize with only a moderate degree of degradation in performance. It is rea-
sonable to suppose that if a set of training data is supplied, better generalization levels can be achieved. Given the lack of a learning algorithm for large neural structures, such as the CoDi neural networks, it is interesting to see that it may be possible to evolve all the necessary features, instead of learning them.

Ultimately, the aim of the CAM-Brain Project is to make artificial brains, using the CBM to evolve large numbers (10,000) of CoDi modules very quickly, and then assemble them into humanly defined artificial brain architectures, such as a controller for a robot kitten and speech processing applications. The CoDi model is powerful enough to evolve "Hubel-Wiesel", line motion detector modules, for example, which is something we have already done.

References


