"CAM-brain" ATR's billion neuron artificial brain project: A three-year progress report

Received: March 2, 1998 / Accepted: April 2, 1998

Abstract This paper reports on recent progress made in ATR's attempt to build a 10000-module evolved neural net artificial brain to control the behavior of a life-sized robot kitten.

Key words Artificial brain · CAM-brain · Artificial neural networks · Evolvable hardware · CAM-brain machine (CBM)

Introduction

This paper reports progress in ATR's artificial brain (CAM-brain) project. The broad aim of this project is to build/grow/evolve an artificial brain containing a billion artificial neurons by the year 2001. The basic ideas of the CAM-brain project are as follows. Use cellular automata (CA) as the foundation upon which to grow and evolve neural network circuits with user-defined functionality. The state of each cellular automata cell can be stored in one or two bytes of RAM. Since nowadays it is possible to have a gigabyte of RAM in one's workstation, there is a huge space in which to store the CA cell states, i.e., more than enough to contain an artificial brain. The next consideration in the CAM-brain project was how to evolve these neural networks quickly enough for "brain building" (i.e., the assemblage of 10000 and more of such evolved neural network modules into humanly defined artificial brain architectures). We chose to use evolvable hardware techniques that our team invented. Special reconfigurable hardware chips (Xilinx XC6264) have been programmed to grow neural circuits literally in nanoseconds, according to a simplified neural growth and signaling model that we call "CoDl." This special hardware will update 100 billion CA cells per second, which is fast enough to complete a full run of a genetic algorithm (e.g., with a population of 100, for hundreds of generations) in about a second. A user-specified fitness definition for each module is compiled directly into programmable hardware in order to automate the grown circuit's fitness measurement at hardware speeds. This hardware is called a CAM-brain machine (CBM), and will be built by the spring of 1998.

Our team is currently involved in several parallel tasks. The first is the construction of the CBM. The second is the behavioral specification and design of a life-sized robot kitten which a 10000 evolved neural network module artificial brain will control. The third is the design of the brain itself. The kitten robot, called "Robokoneko" (Japanese for "robot kitten") will be controlled by an on-body radio link to the artificial brain consisting of some 80 megabytes of RAM, which is updated 2500 times per second by the CBM, which is fast enough for real-time control. The year 1998 will be spent in gaining experience with the CBM and evolving a large number of modules. At the same time, the kitten robot will be built and tested, and the brain architecture will be designed. In 1999, the brain and the kitten should be brought together, hopefully in time to show the world at the turn of the millennium, the first ever functioning artificial brain. The CBM's incredible speed should allow brain building to become practical. Our team hopes that the CBM will give birth to a new research field, that we call simply "brain building." If the kitten robot and its brain are suc-
cessful, then it is likely that industry will be interested. The rest of this paper is divided into the following sections. The next section introduces the neural net model that we are currently implementing in FPGA (field programmable gate array) hardware in a special piece of apparatus we call a CAM-brain machine (CBM), whose description is found in the following section. We then present some early software simulation results of this machine, which, although very preliminary, show the machine and the CoDi model to be highly evolvable. The next section gives some early thoughts on the specifications of the robot kitten, which a 10000 evolved neural net module artificial brain will control. After the summary, the last section gives an idea for a future extension of the CAM-brain project, namely implementing a CBM using WSI (wafer scale integration) technology, which could be two orders of magnitude faster and bigger than the CBM.

The CoDi neural network model

This section describes the neural net model implemented in the CAM-brain machine. We call it "CoDi" (i.e., collect and distribute). CoDi is a simplified CA-based neural network model developed at ATR in the summer of 1996 with two goals in mind. One was to make neural network functioning much simpler compared with the older CAM-brain model developed in 1993 and 1994, in order 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 module, a population of 100 modules is run through a genetic algorithm for about 100 generations, resulting in tens of thousands of different module evaluations. Each module evaluation consists of growing a new set of axonic and dendritic trees which interconnect about 100 neurons in the 3D cellular automata space of 4096 cells, and then running the module to evaluate its performance (fitness). This typically requires 300 update cycles for all the cells in the module.

On the hardware we currently use to update the CA cells, i.e., MIT's CAM-8 machine, it takes one minute to update the twelve billion cells needed to evolve a single neural module. Even in a simple "insect-like" artificial brain, there are a million neurons arranged into ten thousand 100-neuron modules. It would take 7 days (running 24h per day) to finish the computations. Owing to a very large module chromosome size, an evolution time longer than 100 generations may be needed. Another limitation was apparent concerning full brain simulation involving thousands of interconnected modules. For a 10000-module brain, the CAM-8 is capable of updating every module at the rate of five update cycles per second. However, for real-time control of a robotic device, an update rate of 50-100 cycles per module, 10-20 times per second is needed. So the second goal was to have a model which would be portable into hardware which would eventually design a machine capable of accelerating both brain evolution and brain simulation by a factor of 100-500 compared with the CAM-8 machine.

The CoDi model operates as a 3D cellular automata. Each cell is a cube which has six neighbor 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*2 CA cells. Cells are interconnected with bi-directional 1-bit buses and assembled into 3D modules of 4096 cells (16*16*16). Modules are further interconnected with 32-bit wide buses to function together as an artificial brain. These intermodular connections are implemented as linked lists in a module interconnection memory. In a neuron cell, five (of its six) connections are dendritic inputs, and one is an axonic output. A 4-bit 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 neuron's chromosome) and either adds to or subtracts from the accumulator. The neuron cell's output can be oriented in six different ways in the 3D space. A dendrite cell also has five inputs and one output, to collect signals from other cells. The incoming signals are passed to the output with an OR function. An axon cell is the opposite of a dendrite. It has one input and five outputs, and distributes signals to its neighbors. The "collect and distribute" mechanism of this neural model is reflected in its name "CoDi." Blank cells perform no function in an evolved neural network. They are used to grow new sets of dendritic and axonic trees during the evolution mode. Before growth begins, the module space consists of blank cells. Each cell is seeded with a 5-bit chromosome. The chromosome will guide the local direction of the dendritic and axonic tree growth. Five bits encode 32 different growth instructions, e.g., grow straight, turn left, split into three branches, block any growth, or split up or down, etc. Before the growth phase starts, a small number of cells (1-3%) are seeded as neurons at random locations.

As the growth starts, each neuron continuously sends growth instructions 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, which is continuously being sent by a neuron to other blank cells. The direction of the propagation is guided by the 5-bit growth instruction described above. This growth 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.

The CAM-brain machine (CBM)

This section describes the hardware implementation of the above CoDi model, allowing a CoDi neural net module to
be grown in nanoseconds, and a whole population of modules to be evolved in about one second. The CBM consists of five major blocks: (1) cellular automata module; (2) genotype and phenotype memory; (3) fitness evaluation unit; (4) genetic algorithm unit; (5) module interconnection memory.

Each of these blocks is discussed in detail below. The cellular automata module is the hardware core of the CBM. It is intended to accelerate the speed of brain evolution through a highly parallel execution of cellular state updates. The CA module consists of an array of identical hardware logic circuits or cells arranged in two identical 3D structures of 16×16×16 cells (a total of 8192 cells). Cells forming the top layer of the module are recurrently connected with the cells in the bottom layer. A similar recurrent connection is made between the cells on the north and south, and the east and west vertical surfaces. Thus a fully recurrent toroidal cube is formed. This feature allows a much higher axon and dendritic growth capacity by effectively doubling each of the three dimensions of the block.

The CA module is implemented with new Xilinx FPGA devices XC6264. These devices are fully and partially reconfigurable, featuring a new co-processor architecture with data and address bus access in addition to user inputs and outputs, and allow the reading and writing of any of the internal flip-flops through the data bus. An XC6264 chip contains 16384 logic function cells, each cell featuring a flip-flop and some Boolean logic capacity, capable of toggling at a 220 MHz rate. Logic cells are interconnected with neighbors at several hierarchical levels, providing identical propagation delay for any length of connection. This feature is very well suited to a 3D CA space configuration. Additionally, clock routing is optimized for equal propagation time, and power distribution is implemented in a redundant manner. To implement the CA module, a 3D block of identical logic cells is configured inside each XC6264 device, with CoDi-specified 1-bit signal busses interconnecting the cells. Given the FPGA internal routing capabilities and the logic capacity needed to implement each cell, the optimal arrangement for a XC6264 is 8×8×4 (256 cells). Four external surfaces of each module cube have eight cells reserved to be used as external inputs to the module (32 signals total), and two other surfaces have eight cells reserved as external outputs from the module (16 signals). To assemble two identical 4096-cell modules, 32 XC6264 chips are needed in a BGA560 package, laid out as 2×2 chips on eight boards.

There are two modes of CBM operation, namely evolution mode and run mode. In the evolution mode, memory space is used to store the chromosome bitstrings of the evolving population of modules (module genotype). For a module of 4096 cells, 20480 bits of memory are needed. For each module the genotype memory also stores information concerning a maximum of 128 neurons locations and orientations inside the module. This includes, X, Y, Z coordinates (12 bits), gaiting code (3 bits), input functions (excitatory/inhibitory) (5 bits), giving a total of 20 bits per neuron. Thus, the total chromosome memory requirement for one module is 20480 + 20×128 = 2880 bytes. The genotype/phenotype memory size for 10000 modules is 32 Mbytes. The host-computer memory can be used when needed to reload this space with more data.

In run mode, memory is used as phenotype memory for the evolved modules. The phenotype data describe grown axonic and dendritic trees and their respective neurons for each module. For phenotype storage, 6 bits are required per cell, i.e., for gaiting (3 bits), cell type (2 bits), and signal value (1 bit). In addition, neuron cells require 4 bits for the accumulator value stored. Phenotype data are loaded into the CA module to configure it according to the evolved function. Genotype/phenotype memory is used to store and rapidly reconfigure (reload) the FPGA hardware CA module.

Reconfiguration can be performed in parallel with running the module, owing to a dual-pipeline phenotype/genotype register in each cell. This guarantees the continuous running of fast FPGA hardware at full speed with no interruptions for reloading in both evolution and run modes. Thirty-two Mbytes of memory can store over 10000 modules at a time. This is sufficient to evolve 10000 modules at high speed, or to run a simulated brain with one million neurons. A large memory will be based in the main memory of the host computer (Pentium-Pro) connected to the CBM through a PCI (peripheral component inter connect) bus, capable of transferring data at 132 Mbytes/s. Genotype/phenotype memory is connected to the hardware CA module and is used for rapid reconfiguration of the neural module by loading new chromosome data (or phenotype data) into the hardware register of each cell through the XC6264 data bus access. Thus the fast hardware for the CA module is time-multiplexed between multiple neural modules in a large brain.

When a useful module is being evolved, each instance of a module must be evaluated in terms of its fitness for a targeted task. During the signaling phase, each module generates a sequence (array) of 16 output signals, or vectors, which is compared with a target array in order to guide the evolutionary process. This comparison gives a measure of the performance, or fitness, of the module. Fitness evaluation is supported by a hardware unit which consists of an input vector array stack, a target vector array stack, and a fitness comparator. The input stack is 32-bit-wide SRAMs (FIFOs) and the target vector stack is 16-byte-wide SRAMs, storing up to 2048 input and target vectors each to support the signaling phase of up to 2048 cycles. During each clock cycle an input vector is read from its stack and fed into the module's inputs. At the same time, a target vector is read from its stack to be compared with the current module output vector by the fitness evaluation unit. The fitness comparator computes a Hamming distance between each output vector and a corresponding target vector, and accumulates the result for the whole duration of the signaling phase. At the end of the signaling phase, a final measure of the module's fitness is instantly available. Multiple target and input arrays are stored in the host-computer memory.

To evolve a module, a population of 100 modules is evaluated by computing every module's fitness measure, as described above. The ten best modules are then selected for further reproduction. A hardware support will be provided
to store and update a “current best ten” list. This list is an array of the current best module numbers. Each number is an address of the module chromosome in the chromosome memory. After each generation of modules, the ten best are mated and mutated to produce 100 offspring modules to become the next generation. Mating and mutation is performed by the host-computer software on the chromosome memory. Because this process can be performed in parallel with the module evaluation, and only takes place once in a generation, it is expected that there will be no significant slowdown in the evolutionary process. In order to support the run mode of operation, which requires a large number of evolved modules to function as an artificial brain, a module interconnection memory is provided. This consists of an output vector array stack, similar to the input array and target stack, and a 64-Mbyte memory for intermodular pathway storage. When each module of a large brain is configured in the CA hardware core (by loading its phenotype), an input stack is loaded with an array of input vectors. These vectors are previously stored output vectors recorded during the signaling phase of other modules which are connected to this module. A large module interconnection memory will store the connection map and the current state of signals “travelling” between modules. There will be software and hardware support provided for combining output arrays from up to eight modules, to be used as inputs for one module. In addition, an externally connected set of inputs and outputs is provided in the hardware, which will allow the reception of signals from external sensors and the sending of signals to external effectors for robotic control. When running a simple million-neuron brain which contains 10,000, 100-neuron modules, the CBM is capable of updating each module for 100 cycles at the rate of about 25 times per second, allowing real-time control of robotic devices.

Some CBM simulation results

This section presents some early CBM software simulation results. At the time of writing (October 1997), we are very much in the thick of this simulation, so we are unable to present extensive results. However, judging from the results we already have, it is clear that the CBM will be highly evolvable, which is very encouraging. We now give a taste of what we have done. It concerns a digital “timer” module, which will be fairly typical of the type of module which will need to be evolved when the time comes to specify the sensor, control, and behavior modules to be evolved for the artificial brain. We wanted to see if a CoDi 1-bit module (consisting of 4K 3D CA cells, with about 150 artificial neurons in the 4K space) could evolve a “timer,” i.e., where constantly firing inputs generate, at a single cell output (placed elsewhere in the CA space), a string of 0s during the first 30 clocks, then a string of 1s during the next 20 clocks, and finally a string of 0s in the last 20 clocks. We thought this was quite a demanding evolutionary task and would be a useful test vehicle for the early evolutionary trials. The fitness definition was simple. If a 0 appeared in the first (0) block, score one point. If a 1 appeared in the second (1) block, score 2 points. If a 0 appeared in the third block (9) score 3 points. Hence a perfect score would be $30 \times 1 + 20 \times 2 = 130$ points. The population size was 34, with no crossover. The CoDi 1-bit simulation evolved with a fitness of 100% in about 150 generations. Since a perfect score gives no indication of the limits of the evolvability of the CBM, we chose a much more demanding task, which was similar to that described above. Instead of three regions, five were chosen, as shown below, including the best evolved output:

Target output:

```
000000000000000000000000000000001111111111111111
00000000000000000000000000000000111111111111111
00000000000000000000000000000000111111111111111
```

Best evolved output:

```
00000000000000000000000000000000111111111111111
00000000000000000000000000000000111111111111111
00000000000000000000000000000000111111111111111
```

This obtained a fitness score of 96%, where the fitness was defined in a similar way as before, with the point weightings for the five regions being 12, 7, 3, 2, and 1. Since 96% was not a perfect score, it showed the limits of the CBM evolvability. Presumably with more neurons and CA space, greater evolvability will be achievable. Such hypotheses need to be tested in the near future. However, the results so far are rather impressive. It shows that the CBM and the CoDi 1-bit model is capable of evolving modules with quite demanding behavioral characteristics. In the coming months we intend to publish more detailed descriptions of these ongoing CBM simulation experiments. However, we are conscious that for a true test of the capabilities of a CBM we need a CBM.

Robokoneko, the robot kitten

An artificial brain without something to control is rather pointless, so we decided to make the brain control the behavior of a life-sized robot kitten. This section discusses the behavior that the authors would like the robot kitten ("Robokoneko") to show, which will require roughly 10,000 modules to implement. It seems that there are two broad approaches to mapping a set of behaviors to a given number of modules, and vice versa. One approach is empirical, i.e., just keep adding behaviors and capabilities to the brain until the number of modules concerned is reached, or just add more modules until a given behavioral repertoire is reached. The other approach is more theoretical and does not yet exist. It is called "behavioral set analysis" (BSA), and attempts to provide brain architects with principles with which to estimate the number of modules needed for a given behavioral set, or for a given number of modules, how extensive (how intelligent?) the behavior of such a brain could be. Since brain building is entirely new, at least on the scale envisioned in this paper, BSA does not yet exist. There is not yet sufficient experience to create such a body of theory, but as more artificial brains are built (as CBMs
and similar devices become more commonplace), this theory should be easier to construct. Since the attempt to build a 10000-module artificial brain is the first of its kind, there is no experience to fall back on. It is possible that the behavioral set suggested here may need a lot fewer than 10000 modules. de Garis implemented an artificial creature with less than 100 modules which performed “intelligent” behavior such as stalking prey and mates, and avoiding predators. So imagine what one could do with 10000 modules (possible in 1998), or 1000000 modules (which may be only a few years away). We now give some suggestions as to what Robokonoko could do with the 10000 modules that the CBM will be able to evolve and control in real time in 1998. Note, that this paper only presents the early thoughts of the authors. More detailed plans must be created by the end of 1997 in anticipation of delivery of the CBM in the spring of 1998. The behavioral set of the kitten robot is partly determined by “political” considerations. Building an artificial brain is a pioneering effort, and so our first task is to show the world that such a thing is possible. ATR's brain builder group needs “proof of concept.” For this reason, it is important to make the brain as media-friendly as possible to maximize publicity. When millions of people see a cute media-friendly robotic kitten displaying the many behaviors made possible by a 10000 module artificial brain, they will be impressed, and hopefully so will Japan's top scientific research policy makers who control the purse strings to pay for 2000 evolutionary engineers (EE) to work on a 5-year 1-brain project (2000-2005). Hence, the broad aim is to create an artificial brain which will be capable of making a kitten-sized robot which is as life-like as possible. Since 10000 modules is a lot, it will be possible to give this robot many behaviors and capabilities. After a 2-day brainstorming session the following desiderata for the robot were produced. (A few more aspects were added later.) The body of the robot: kitten should be real kitten size, with, of course, four legs, a tail, whiskers, a head, two color-camera eyes, two microphone ears, and maybe (if it is not too heavy) a soft furry covering over its metal springs and frame, to make it cute. The kitten should be controlled by wireless modem and be battery powered to be active for about 30 min before needing a recharge. The robot could probably be built within a year, especially if graduate students are involved. For functionality, we thought of the following list. For behavior, we wanted the kitten to walk on its four legs, stand on three legs, manipulate with its front leg(s), jump on the floor, jump onto objects, curl up, lie down, sleep, move its tail, move its head, wiggle its ears, react to whisker touchings, urinate (water), avoid obstacles, purr when stroked, show hostility when its tail is pulled, etc. This is only an initial list, and will probably be added to extensively considering we have 10000 neural net modules at our disposal to put into the kitten's brain. For sound generation and detection, we want a speaker with a ROM “meow” expressing a range of emotions, e.g., attention seeking, anger, sadness, purring, etc. The sound detector would be two microphone ears and two signal processing chips, so that one could call the kitten “Here, kitty kitty kitty,” “Scram,” “Woof” (fear reaction), etc.

We wanted heat sensors (to detect people in the neighborhood), touch sensors (on the back, whiskers, tail, etc.) and feet force sensors. For vision, we wanted two color cameras for the eyes, with accompanying signal processing chips on the robot. The whole thing should weigh about 1-2 kg, so that it can jump and move at real kitten speed. It will be made as light as possible and with minimal onboard processing. Most of the intelligence of the kitten will be generated off-body by the artificial brain, whose decisions will be transmitted by wireless modem to (and from) the kitten. A certain minimum weight will be necessary due to the weight of the modem (about 0.5 kg) and the battery (0.5 kg). The binary I/O of the modem could interface directly with the 1-bit signaling of the “CoDi model” of ATR's CAM-brain design.

Summary

This section gives a brief summary of the ideas expressed above. The CAM-brain project hopes to build a billion-neuron artificial brain by the year 2001. The bottleneck in achieving this figure will not be the technology - we have already put 10 million artificial neurons into one huge circuit - but rather, designing the architecture of the artificial brain. More realistically, we will be happy if we can build an artificial brain with 10000 modules (i.e., about a million artificial neurons) to control a robot kitten we call Robokonoko. If this goal can be achieved by the turn of the century, then it is likely that a scaled-up CAM-brain project could aim for a billion-neuron artificial brain within a year or two by adding many researchers and EEs to the team. To evolve the neural net modules quickly, the CoDi 1-bit neural net model presented above will be implemented in special FPGA-based hardware that we call the CAM-brain machine (CBM), which will update CA cells at the rate of about 100 billion a second, allowing a complete run of a genetic algorithm in about a second. The CBM will allow brain building to become practical. However, since greater speed is always essential, we have plans to increase evolution speeds by another two orders of magnitude. The timetable for completing the various remaining tasks to make the kitten robot and its brain as follows. The kitten's behavior and architecture are to be decided before the end of 1997. Work on building the kitten should start by April 1998. The CBM should be built by the spring of 1998. Once we have the CBM in hand, we can quickly evolve modules and gain experience in what the CBM can do and what types of modules are evolvable. This experience will help in a major way when designing the 10000-module artificial brain to control the behavior of the kitten robot. This brain will be designed and its modules evolved throughout the year 1998. The year 1999 will be spent in integrating the kitten robot and the artificial brain.

Now that the CBM is expected to be completed by the spring of 1998, we have started thinking about the next step, namely the possibility of using WSI techniques to achieve
greater performance levels. Our initial major assumption is that applying E-Hard (evolvable hardware) techniques to the wafer would result in the evolution adapting to the inevitable fabrication faults of the wafer. The wafer's much greater area compared to a chip could then contain one huge circuit, consisting of all four components of E-Hard, i.e., the chromosomes used to evolve the circuit, the circuit itself, the hardware-compiled fitness definition, and the GA management. Wafer-scale E-Hard would be the ultimate in speed. The next assumption concerned the diameter of the wafer. A state-of-the-art wafer is 12 inches, at 2.5 cm per inch, and considering that a Xilinx XC6264 chip has a surface area of roughly a square centimeter, that means that the wafer should be roughly 750 times greater in area. The CBM consists of about 50 chips of 1 cm² each, so the wafer should be about 15 times greater in area than the CBM. Implementing a wafer would imply using ASIC (application-specific integrated circuit) techniques, which are about four times denser than FPGAs. Clocking at about 30 MHz and with 9000 CoDi cells in the CBM implies about 15 × 40000 CoDi cells in the wafer, i.e., 500,000. (Of course this is an upper bound, because space would be needed for the other three E-Hard components - the chromosomes, the fitness definition, and the GA control.) The total CA cell update rate of a wafer-CAM would be 500,000 CoDi cells at 30 MHz, i.e., 15 trillion/s, or some 150 times faster than the CBM. The greater area and speed would probably allow the chromosomes to be grown and their fitness measured in parallel. If building this wafer is not too expensive, we would like to investigate the possibility of applying the CoDi model to WSI, particularly now that that WSI is commercial. (There are now WSI companies in Silicon Valley.)

References

(Note: Any paper containing the name de Garis, can be downloaded from his web site at URL: http://www.ship.ac.jp/~degaris)


