Integrating the BSP400 neurocomputer with the MetaNet Network Environment

Jacob M.J. Murre, Jan N.H. Heemskerk, Steven E. Kleynenberg and Patrick T.W. Hudson
Leiden University, Unit of Experimental and Theoretical Psychology
P.O. Box 9555, 2300 RB Leiden, The Netherlands
E-mail: MURRE@HLERUL55.Bitnet
Tel: (31) 71/273631, Fax: (31) 71/273619

ABSTRACT

This article describes the integration of hardware and software components into a single neural network environment. The building blocks described here are the BSP400, a massively parallel neurocomputer and MetaNet, a general neural network design tool. This research is a feasibility study for future developments in large scale integration of computational resources. The objective is to obtain a neural simulation environment that supports easy development and efficient parallel calculation. Integration takes place using device drivers. Apart from handling communications between the components the primary task of the device driver is the mapping of a neural network definition onto the processors of the parallel hardware such as the BSP400. This mapping problem is caused by hardware constraints, such as maximum amount of connections and the topology of the processor network (e.g., transputers in a grid topology). Ultimately we aim at an environment in which the user can program a single 'ideal virtual' machine, which in reality is assembled from many different computational resources.

1. Introduction

Here we describe the integration of the MetaNet Network Environment [8][9] with the massively parallel hardware of the Brain Style Processor (BSP400) [2][3][4]. The objective is a system for solving real-world problems with neural network paradigms. In our approach neural networks serve as virtual machines that can be 'programmed' to solve problems. Programming a network is done by setting the appropriate weights (fine-grained programming) or by broadly prestructuring a network (coarse-grained programming) and using a learning algorithm to arrive at the solution. To achieve this we focus on two principal problems: (1) the interaction problem: 'How can we program a neural network?' and (2) the integration problem: 'How can we effectively use massively parallel hardware to run neural networks?'. In our view, solution of these problems is attainable only through combined research efforts in hardware (neurocomputers), software (neurosimulators), and neural network theory (modelling and application).

(1) The interaction problem. A direct access to all aspects of the neural network at hand is crucial to the successful solution of most problems. Visualization and manipulation of network objects in a design sensitive environment enable the user to think about the problem in terms of neural networks. A full programming cycle includes prototyping, development, simulation, monitoring, and perhaps maintainance. Each of these phases must be supported by an adequate development environment. Without this the general applicability of neural networks is highly doubtful. We have developed such an environment, called the MetaNet network environment which will be described in section 2.

(2) The integration problem. For any but the most trivial applications neural networks must eventually be run on parallel hardware to provide workable solutions. Therefore, in 1988 we initiated a project aimed at building a massively parallel machine. This was done in cooperation with the Electrical Engineering Department at the TU Delft. This machine, the BSP400 (Brain Style Processor with 400 processors) is developed as a feasibility study for much larger versions [5]. It will be described in section 3. Several aspects of a scaled-up version are being investigated, including limited precision implementations of neural networks, VLSI implementation and architectures for very large machines.

Though in the present article we focus on the BSP400 neurocomputer, our research is in fact concerned with the integration of various computational resources (i.e., neurocomputers) into a single system. Device drivers form a crucial part of the interaction between MetaNet and hardware. They will be described in section 4.

CH 3065-0/91/0000-1476 $1.00 © IEEE

2. Introduction to MetaNet

The Connectionist Research Group at Leiden University has carried out research in the field of Learning Neural Networks since 1987. These investigations resulted in the development of CALM, the Categorizing And Learning Module [10][11]. CALM is a building block for modular neural networks, it categorizes and learns arbitrary input patterns. Multi-module CALM networks have been applied to a variety of tasks such as handwritten character recognition, speech-synthesis and motor-contol.

At the time we developed CALM, no environment was available that could adequately handle modular neural networks with a non trivial internal structure. We therefore, decided to develop our own system: the MetaNet Network Environment.

2.1 The MetaNet approach

MetaNet is conceived as a neurosimulator that is consistently extensible with novel network paradigms and parallel hardware. When 'programming' a neural network, user-interaction should be *problem oriented*. Visualization and manipulation of network objects in a design sensitive environment enable the user to think about problems in terms of neural-networks, rather than the specifics of the hardware used or the idiosyncrasies of some standard, linear programming language (such as C or LISP). The MetaNet user should be able to run a neural network on various types of hardware without making alterations in the defined network. Conversion to instructions for a parallel machine in done automatically. If novel hardware is added to the environment, all network paradigms should be supported, preserving MetaNet's independency from the computational resource used. Summarizing, the primary objective of our research is to further develop MetaNet up to a point where it meetss the following three demands:

(1) A problem oriented user interface.
(2) Consistently extensible with novel paradigms
(3) Independency of computational resources

In the current state MetaNet partially incorporates these features. The user interface is build on familiar design concepts, it is menu-driven and graphics-oriented. The user defines networks in a essentialy unbounded two dimensional workspace, in which he can scroll and zoom, allowing him to examine the workspace in any detail, from any viewpoint. At the moment a fixed hierarchy of the network objects (called Problem Oriented INteraction Tools or POINTS), is imposed on a network topology. Ascending from the lowest level this hierarchy consists of NODES, GROUPS, MODULES and MODELS. These can be linked by TRACTS and CONNECTIONS, where a TRACT is a set of CONNECTIONS. Specific neural paradigms are entered at the MODULE level, it is at this level MetaNet decides whether computation is to be performed internally or on external hardware. If there is a device driver available in MetaNet's resource map, that can handle a particular neural paradigm, this will automatically lead to external computation. The user however is completely shielded from all the intricacies involved, see Figure 1.

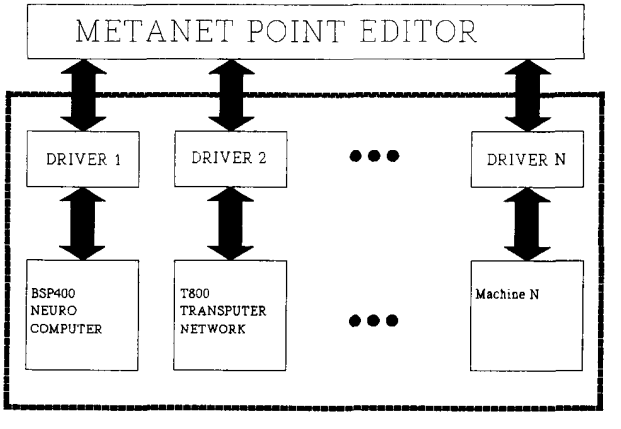


Fig.1. Global overview of the MetaNet Network Environment.

2.3 MetaNet and external parallel hardware

MetaNet is currently being developed for MS-DOS machines and will be soon ported to VAX-VMS and other operating systems. Communication with external (parallel) hardware is accomplished by device drivers, a distinct class of programs, that is part of the operating system, rather than the MetaNet User Interface. A novel piece of hardware can be added to the environment if a device driver is available. This approach preserves the objective that reconfiguration of the environment can be accomplished without modifications in the MetaNet software. On the other hand it should be clear that each novel piece of hardware must be supported by a device driver, that must conform to MetaNet standards. MetaNet is supported by a generic device driver, which can be extended to the requirements in any type of hardware. Multiple device drivers can be installed at system startup. The combination of MetaNet, device drivers and parallel hardware forms a system that is presented to the user as a virtual machine. Further detail of this approach can be found in [9].

3. Introduction to the BSP400

The Brain Style Processor with 400 simple microprocessors (BSP400) forms a massively parallel multiprocessor system. Implementing a neural network on the BSP400 is at node level. This means that each Local Processing Unit implements one specific node in the network.In developing the Brain Style Processor, we focussed on *number* rather than *speed* of the individual processors. The idea is that a real brain style machine inherits some of the properties of the brain, such as graceful degradation. In order to cope with the connectivity problem in large multi-processor systems, a modular design was used. The brain also seems to follow a modular design principle, both on a macroscopic level (e.g., hemispheres, lobes, etc.) and on a microscopic level (the minicolumns in the cortex e.g., [6])). The principle of modularity ensures a scalable design, which we have formalized in a novel hardware concept called the Fractal Architecture [2]. *Scalability* is one of the main design issues in neurocomputer building. It means that the number of processing elements in the system can be increased without deterioration of the overall performance. To implement scalability, the BSP400 system was designed as a number of Units connected to a Communication-bus. The system can be extended - up to a certain limit - by simply adding Units. These Units can be compared to modules in neural networks. The BSP400 was primarily developed for the implementation of the CALM model and other types of biologically plausible modular neural networks. CALM networks consist of a number of modules, which are not all totally interconnected. Such restrictions on connectivity can be shown to have a number of practical advantages [12].

3.1 The BSP400 architecture

A global overview of the BSP400's architecture is shown in Figure 2. The system consists of a Host computer, a Network Controller, up to 31 Units and a Communication-bus that connects all Units and the Network Controller.

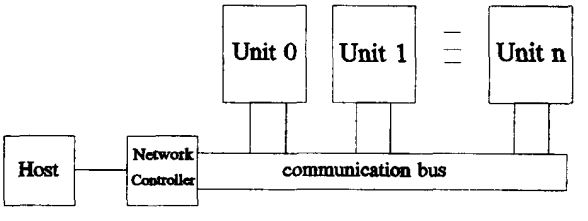


Fig.2. The Brain Style Processor (BSP400).

The *Network Controller* transfers data and commands between the Host computer and the Units over the Communication-bus. Furthermore, the Network Controller synchronizes the calculations in the nodes

on the Module boards, and monitors the state of the neural network. This makes it possible to report special states (e.g., convergence on module x) immediately to the Host.

The *Units* in Figure 2 stand for either *Module boards* or *I/O boards*. All Module boards contain 16 processors that can be considered as Local Processing Units (LPU's). The current version of the BSP400 consists of 25 Module boards. I/O boards serve to directly transfer activations between sensors and motors to and from the system. This direct activation exchange makes the BSP400 suitable for real-world interaction. Module and I/O boards are fully transparent to the Network Controller. They are interconnected by the system wide Communication-bus. They are completely interchangeable and may be added or removed with no consequences for the rest of the system.

3.2 Distribution and calculation processes on the BSP400

The calculation process on the BSP400 is controlled by the Network Controller. Once the Network Controller has been initialized through the Host computer the system functions 'stand alone' (unmediated by the Host). The calculation process can be divided into three phases:

(1) *External distribution.* This is the exchange of activations among the Units over the Communication-bus. The Units are sequentially prompted to put their activation values on the bus, which in turn are read by (possibly) all Units in the system. Activations are read out from the bus by all selected Module boards in parallel. Special unit select circuitry on the Network Controller and Input Activation Queues (IAQ) on the Module boards ensure fast activation transfer in the external distribution phase.

(2) *Internal distribution.* This is the distribution of activations from the IAQ to the individual nodes on a Module board. All Module boards operate in parallel. The functioning of the IAQ is fully asynchronous, so the Network Controller can use the Communication-bus for various purposes during the internal distribution for example to collect specified activations to be sent to the I/O boards or to monitor of activations for convergence checks. The old activations remain accessible till the end of phase 3.

(3) *Calculation of activation and learning rule.* The node software currently implemented contains activation and learning rules of the CALM model [10][12][1]. While the nodes are calculating *new* activations, the Network Controller may still access *old* activations of the previous calculation cycle. At the end of phase 3 the new activations are set to the outputs of the nodes. When the learning rule takes up a great part of the neural calculation, new activations may be set at the outputs and distributed through the system while the nodes calulate their learning rule. This can be done with all paradigms that do not depend on a separate training and recall phase. It reduces the total iteration time by parallelizing communication and calculation.

It is possible to send batch jobs to the BSP400. This is useful for training purposes. A set of input vectors can be presented together with convergence instructions (e.g., what to do when certain modules converge) to the Network Controller. The Network Controller reports to the Host what is going on in the system and may replace the current input vector by the next.

4. Integration of MetaNet and the BSP400

To effectively integrate MetaNet with the BSP400 through a device driver we had to consider the fact that the BSP400 was designed as a standalone machine which, after initialization, can run without further external interference. To preserve this feature we aimed at a device driver which can communicate with the BSP400 either directly or in a background process (remotely). From the perspective of MetaNet this strategy greatly enhances the freedom with which the user can interact with the environment. After the user has defined a network topology, network calculations can be performed. In the meantime the user can do other tasks, requesting the computational results at a convenient moment.

MetaNet allows integration of multiple forms of different external computational resources, each communicating through its own device driver. The BSP400 device driver currently implemented in our environment serves as a generic prototype for future generations [13]. It has a modular structure consisting of independently modifiable sections, that parse input from MetaNet and signal and correct errors in the communication flow. The device driver translates MetaNet commands into machine specific instructions, and verifies whether the computational requests are within the scope of the BSP400's capacity. It also tries to find the mapping of the neural network definition to the Local Processing Units of the BSP400. If all checks have been passed, the driver moves the BSP400 instruction-code into a transmission buffer which is then sent to the BSP400 on interrupt basis. Through interrupt-driven process calculation may take place in a background task. The device driver also translates BSP400 output into

the predefined format in which MetaNet requests its data. This mirror process is also executed as a background task. Should at any point in this bidirectional flow an error occur, the device driver will try to correct it (e.g., by re-iterating the question), or in case an external system goes 'down', by gracefully terminating all related processes within MetNet.

5 Concluding remarks

The current integration of the BSP400 and MetaNet serves as a feasibility study for future neurosystems. These machines will be larger and more complicated. The next generation of the BSP400 will support 40,000 or possibly even 4,000,000 processing elements which can handle different types of modular networks. MetaNet will be further extended so that it will be able to integrate different types of neurocomputers. Our research on MetaNet currently focusses on the development of novel drivers (for the Connection Machine, VAX Vector Processor and for transputer networks [7] and on the definition of a formal description of neural networks in terms of process dependency graphs [9].

References

[1] Heemskerk, J.N.H., and F. Kleer (1989). *Learning neural networks in digital hardware.* (In Dutch: 'Lerende neurale netwerken in digitale hardware'.) Internal report, Leiden University, Unit of Experimental and Theoretical Psychology.

[2] Heemskerk, J.N.H., J.M.J. Murre, J. Hoekstra, L.H.J.G. Kemna, and P.T.W. Hudson (1991). The BSP400: a modular neurocomputer assembled from 400 low-cost microprocessors. In: O. Simula (Ed.) *Proceedings of the International Conference on Artificial Neural Networks - ICANN-91-Espoo, Finland,* in press.

[3] Heemskerk, J.N.H., J.M.J. Murre, J. Hoekstra, L.H.J.G. Kemna, and P.T.W. Hudson (submitted). *The BSP400: Towards a large modular neurocomputer.* Submitted to IEEE Transactions on Neural Networks.

[4] Hoekstra, J.,J.N.H. Heemskerk, A.J. Klaassen, R.H. Phaf, P. Knoppers and P.T.W. Hudson (1990). Hardware design concepts for a CALM neural network using 400 simple processors: system architecture. In: B. Widrow, and B. Angeniol (Eds.) International Neural Network Conference, INNC-90-Paris, 676. Dordrecht, Kluwer.

[5] Hudson, P.T.W., J.M.J. Murre, and R.H. Phaf (1990). *Neural networks for solving real-world problems in massively parallel hardware: a unified approach to user-interaction and system-integration.* SION Research Program 612-322-018.

[6] Mountcastle, V.B. (1978). An organization principle for cerebral function: the unit module and the distributed system. In G.M. Edelmann and V.B. Mountcastle (Eds.). *The mindful brain.* Cambridge, MA: MIT Press.

[7] Murre, J.M.J. (1991). Transputer implementations of neural networks: an analysis. In O. Simula (Ed.). Proceedings of the International Conference on Artificial Neural Networks, ICANN-91-Espoo, Finland.

[8] Murre, J.M.J., and S.E. Kleynenberg (1990). The MetaNet environment for the development of modular neural networks. In: B. Agneniol, and B. Widrow (Eds.) *Proceedings of the International Neural Network Conference, INNC-90-Paris,* 717-720. Dordrecht: Kluwer.

[9] Murre, J.M.J., and S.E. Kleynenberg (1991). Extending the MetaNet Network Environment: process control and machine independence. In: O. Simula (Ed.). *Proceedings of the International Conference on Artificial Neural Networks - ICANN-91-Espoo, Finland,* in press.

[10] Murre, J.M.J., R.H. Phaf, and G. Wolters (1989). CALM: A modular approach to supervised and unsupervised learning. IEEE-INNS, Proceedings of the International Joint Conference on Neural Networks, Washington DC, 1, 649-656.

[11] Murre, J.M.J., R.H. Phaf, and G. Wolters (1990). *Neuronal data processing network.* Internation Patent Application nr.PCT/NL90/00018.

[12] Murre, J.M.J., R.H. Phaf, and G. Wolters (1990). *CALM: Categorizing And Learning Module,* Psychological Reports, ETP 02-90, Leiden University, Leiden, (also submitted).

[13] Oordijk, A.H., and R. Westgeest (1991). *System integration of the MetaNet and BSP400 through MS-DOS device drivers.* Internal report, Leiden University, Unit of Experimental and Theoretical Psychology.