Massive parallelism for Artificial Intelligence
(extended abstract)

Luc Steels
VUB - Artificial Intelligence Laboratory
Pleinlaan 2, 1050 Brussels Belgium

There is at the moment an explosion of research attempting to explore massive parallelism for Artificial Intelligence. The paper reviews these efforts. This is an extended abstract indicating the major topics that will be discussed.

It was already clear in the early seventies that parallelism was going to be necessary for Artificial Intelligence. Initially research concentrated on small scale parallelism. It was assumed that there would be a gradual scale up of the number of processors with a factor of 10 or at most 100 and that the programming media used to program such networks of processors were to be relatively straightforward extensions of existing programming languages.

Since then machines have been conceived and built with these characteristics. Examples are the BBN butterfly and the INTEL hypercube. Each elementary unit in these machines is itself a fairly complex microprocessor (e.g. a Motorola 68000) with a fair amount of memory. Nodes are connected in a network that supports open-ended communication: Every node can in principle communicate with every other node.

At the same time various AI researchers conceived programming languages to program small-scale parallelism. These language design efforts which are mostly concurrent versions of existing AI programming languages fall in three main categories:

1. ACTOR LANGUAGES: The ACTOR paradigm of computation emerged out of work on object-oriented, message passing languages such as Smalltalk. It proposes active objects that perform computation by passing messages. This paradigm was already conceived in the early seventies (Hewitt, 1977) and several languages and simulations were built

(see e.g. Lieberman, 1981, Yonezawa, 1983).

2. FUNCTIONAL LANGUAGES: LISP was developed in the early sixties and emerged as the major paradigm for symbolic programming. It is therefore natural that people started to develop parallel versions of LISP. An example of this is a concurrent Common LISP developed for the INTEL hypercube (Barber, et.al. 1986) or the BBN parallel LISP developed for the BBN Butterfly computer.

3. LOGIC PROGRAMMING: PROLOG emerged in the early seventies as a valuable alternative symbolic programming paradigm. Intensive research has since been carried out to develop parallel versions. Examples are PARLOG (Clark 1985) and Concurrent PROLOG (Shapiro, 1984). A concurrent logic programming language is also one of the targets of the Japanese 5th generation computer project.

Although all these developments have attracted a considerable amount of attention and enthusiasm, they nevertheless do not appear to give the very large scale up that we could potentially get from parallelism. In particular they only increase the efficiency of existing models of intelligence (e.g. search will be done a little bit more quickly) but there is no qualitative difference in the models.

But while these experiments were going on another development started to take place. This development grew out of research on semantic networks and particularly research on new architectures to support parallel operations over semantic networks such as the propagation of markers to compute set intersection (Fahlman, 1978).

By the early eighties a first example of such an architecture, called the connection machine, was fully engineered and commercially available (Hillis, 1986).

The main characteristic of this machine is that it supports massive parallelism: there are tens of thousands of processors although each processor is relatively simple. It was quickly discovered that although the architecture may have been inspired initially by semantic networks, a lot more could be done with it. In particular it is a general purpose SIMD with a large number of additional features, particularly related to flexibility of programming from higher level languages. Consequently the machine supports well data level parallelism: the execution of the same operation over large amounts of data. New programming media, such as *LISP and CM-LISP (Hillis and Steele, 1986) have been developed to program the machine from this perspective. Many applications have been worked out in such areas as physical simulation using cellular automata (Wolfram, 1983), document retrieval (Waltz, 1986), simulation of other architectures (), etc. The first part of the paper introduces this development in some more detail.

At the same time radically different models of intelligence started to emerge which required the kind of massive parallelism supported by connection machine type architectures. The first development grew out of neural network models which were already developed in AI in the early sixties but abandoned due to apparent fundamental limitations (Minsky and Papert, 1968). New theoretical ideas together with the availability of parallel machines enabling experimentation have given new impetus to connectionism and it is now one of the most active research areas within Artificial Intelligence (Rumelhart and McClelland, 1986). The second part of the paper contains a review of these developments.

The second development grew out of work on cellular automata. Cellular automata were originally developed by von Neumann. They have shown a steady development particularly from a theoretical point of view (Toffoli and Margolus, 1986). It was also discovered that many phenomena

in the physical and biological sciences related to complex dynamic systems could be fruitfully studied with the aid of cellular automata.

(Wolfram, 1983). Models based on cellular automata are also beginning to be applied to Artificial Intelligence.

One example of this is a model of subsymbolic

computation where symbols or rules are represented as groups of particles moving in computation space (Steels, 1987a). One of the interesting features of this approach is that phenomena of self-organisation which have been studied in the context of physical or biological systems (Nicolis and Prigogine, 1978, Babloyantz, 1986) can now be shown in AI system (Steels, 1987b). The final part of the paper elaborates on these developments.

REFERENCES

- Babloyantz, A. (1986) *Molecules, Dynamics and Life. An introduction to Self-organization of Matter*. Wiley-Interscience New York.
- Barber, G. et.al. (1986) *A parallel LISP for the INTEL Hypercube*. Gold Hill Computers.
- Clark, K. (1985) *PARLOG*. Technical Report. Imperial College. London.
- Fahlman, S. (1978) *NETL: A system for representing and using common-sense knowledge*. MIT Press Cambridge, Ma.
- Hewitt, C. (1977) *Viewing control structures as patterns of passing messages*. Artificial Intelligence 8.
- Hillis, D. (1986) *The Connection Machine*. MIT Press. Cambridge, Ma.
- Hillis, D. and G. Steele (1986) *CM-LISP*. Proceedings of the LISP and functional programming conference. Cambridge Ma.
- Lieberman, H. (1981) *Thinking about lots of things at once without getting confused: Parallelism in Act-1*. AI Memo 626. MIT AI Lab.
- Minsky, M. and S. Papert (1969) *Perceptrons*. MIT Press, Cambridge, Ma.

Nicolis, G. and I. Prigogine (1977)
Self-Organization in Nonequilibrium Systems.
Wiley-Interscience, New York.

Rumelhart, D. and J. McClelland (eds.)
(1986) Parallel Distributed Processing:
Explorations in the Microstructure of
Cognition. Cambridge, Bradford books.

Shapiro, E. (1985) A subset of concurrent
PROLOG and its interpreter. Weizman
Institute. Technical Report.

Steels, L. (1987a) Introduction to sub-
symbolic computation. VUB AI lab Memo 87-5.

Steels, L. (1987b) Self-organisation through
selection. Proceedings of the first IEEE
neural network conference. San Diego.

Toffoli, T. and N. Margolus (1986) Cellular
Automata Machines. MIT Press, Cambridge, Ma.

Wolfram, S. (1983) Statistical mechanics
of cellular automata. Reviews of Modern
Physics. Vol 55, no 3.

Yonezawa, A. (1986) ABCL. Technical Report.
Tokyo Institute of Technology. Tokyo.