How Can Slow Components Think So Fast?

Stephen M. Omohundro
Department of Computer Science and
Center for Complex Systems Research,
University of Illinois at Urbana-Champaign,
508 South Sixth Street, Champaign, IL 61820, USA.

February 2, 1988

Most actions are stored rather than computed. This paper's answer to the question posed by the symposium title is that most intelligent behavior is accomplished without much thinking. It therefore doesn't take much time to accomplish even with slow components. In some ways this is a rephrasing of what has almost become an AI cliché: "Knowledge is power". This phrase is meant to emphasize the importance of stored knowledge over fancy inference procedures. If most actions in biological systems are essentially replaying rather stereotyped stored procedures, then the speed issue becomes much less problematic. The greatest speed is required in real time interactive tasks and if the appropriate cached response has been stored and its calling circumstances can be recognized, it may be played back with very little delay. It may be that such an approach is essentially forced on a system with slow components. The effect on the architecture is to optimize things for effective learning (either by the organism or during evolution) and around the task of recognizing the best action for the current context.

In lower animals, such as the cockroach, the appropriate action for an organism to take in a given context is genetically built directly into the organism's nervous system. The nervous systems of these animals as well as the reflex arcs of higher organisms achieve great response speed by having simple networks connecting sensory inputs to appropriate motor ganglia. When a cockroach's wind sensor is stimulated (presumably by the approach of a predator), it directly initiates an appropriate turning response unless inhibited by higher centers. There are only a few neurons between the sense organ and the effector and for greater speed these often have thickened axons. The behavioral information at this level is rather directly encoded in the pattern of connection between neurons. Higher organisms are more adaptive and use networks with large modifiable portions between input and output.

Brain architecture. Modern neurophysiology is making rapid progress in determining the anatomical and functional architecture of the brain and nervous system [5]. While there is much left to understand, one of the most fundamental features of the emerging picture is that the brain is made up of a number of functionally different areas joined together by ordered bundles of interconnecting fibers. The cerebral cortex is quite structured and much progress has been made in mapping its interconnection structure. In 1909 Brodmann decomposed the cortex into 52 different areas based on subtle anatomical differences in the size and density of cells, the layer structure, and the density of axons innervating each region [1]. More recent work has shown that the partition defined by Brodmann's areas corresponds well to partitions defined both by distinct functional behavior and by the innervation of individual bundles of interconnecting fibers. These more recent studies indicate a slightly finer decomposition than Brodmann's but the total number of areas is estimated to be at most about 200 [2].

Much progress is being made on mapping out the interconnection pattern of these areas. For example, studies of the visual system of the macaque monkey have identified twelve areas split into two major channels, one specialized for motion perception and one for form perception [3]. Each area roughly preserves the spatial layout of the retina, the interconnection graph of the areas has a hierarchical structure in which the modules fit naturally into six successive layers, and most areas have inputs and outputs to only one or two others. V2, the area with the largest number of outputs innervates five other areas and MT, the area with the largest number of inputs is innervated by four other areas. In [6] the New World monkey is described as having ten visual areas and the macaque as having seven somatosensory areas, and six auditory areas. These areas all tend to be topographically structured. The more complex systems appear to have evolved by introducing more cortical areas. An early animal like the hedgehog apparently has only two visual and two somatosensory areas.

The type of computation that can be performed in a brain is severely constrained by the limitations of neurons. The total time to perform an interesting computation such as recognizing an object is about 0.5 seconds, while an individual neuron is only capable of firing in time intervals of about 0.005 seconds. Therefore only about 100 layers of neurons from sensory input to motor output can be involved in such a computation [4]. This limit, together with physiological features described above suggests that intelligent computations can be performed in networks consisting of less than 200 modules, each of which performs a function that can be accomplished within a few layers of neurons. There are probably at most 20 to 40 modules along any path from input to output. Each module, while quite restricted in depth by the speed of neurons, may be quite wide and can perform much of its computation in parallel. There are estimated to be about 10^{11} neurons in the human brain.

Lower level functions are parallel. In early animals information flowed directly from sensory input to motor output. In higher animals this same basic flow still exists but several higher levels control it. The representation of information near the periphery is rather direct and highly parallel. The visual information on the retina is conveyed on a topographically mapped bundle of about 10^6 fibres and the firing of a single neuron is directly correlated to sensory information in a given region. A similar statement holds for the auditory and somatosensory maps, as well as for those neurons involved in motor control. Each of these peripheral areas is able to perform a simple set of prespecified highly parallel operations under higher level control. The different areas are non-interacting at this stage and may be thought of as implementing separate processes in MIMD fashion. Within each area, many separate columns perform the same operations on different portions of the input under the control of fairly simple signals from higher levels. These regions may be thought of as analogous to SIMD machines.

It will be very important to identify the operations available from these highly parallel modules. In reference [10] psychophysical data is used to try to infer the basic "visual routines" into which high level visual cognition tasks are compiled into. A number of candidate operations are also suggested by neurophysiological data. Early the visual pathway there are modules which do some kind of edge detection, color extraction, lightness compensation, and motion extraction. One very useful task would be to return the location of the unit of a particular type which is firing most strongly. This would be a neurophysiological correlate to our ability to pick out and focus attention on the brightest spot in an image. Similarly we can in parallel pick out a red dot in a green field, or an X in a field of O's or a pin prick on the body's surface. Limitations on performance of these tasks (conjunctions like finding red X's require serial scanning) should give us insight into the underlying operations.

Higher level functions are serial. There is much psychological evidence that the higher level control is essentially serial in nature and runs on the exceedingly slow clock cycle predicted by neural limitations. For example, children count objects a little more slowly than one per second. Studies of performance on reasoning tasks are consistent with the estimate that a single high level "procedure call" takes about 50 to 100 milliseconds (about 10 or 20 neuron firings). In perceptual studies, one finds the phenomenon of "focus of attention" in every modality. It appears that the higher level centers must choose a particular portion of a particular sensory input to work on. In contrast to the operations at the periphery, the high level system appear quite general and is capable of learning arbitrary relationships and performing arbitrary operations.

The high-level system has access to a declarative memory in which it may store and retrieve items at will (subject to hardware capabilities). This memory must be capable of associative retrieval based on partial information. In the early stages of performance of a task, it may run in "interpreted" mode in which the steps of a procedure are stored declaratively and carried out with error checking and reality testing. There appears to be a separate "procedure" memory which can be used to quickly carry out long strings of prestored actions. These actions are built up in chunks over long periods of practice using interpreted actions from the declarative representation. It would be extremely dangerous for the system to be able to directly modify the procedural memory for it could insert code saying things like: "don't do anything ever again". The process of "chunking" useful procedural elements seems to occur at a universal rate througout the system [8] and may therefore occur everywhere by using the same mechanism.

Learning inhibits parallelism. If stored information is central to an organism's functioning, its architectural design and representation schemes should be organized to build it up very efficiently. This may be one reason why the high level operations are serial. It is much easier to modify procedures (as one must do in learning and evolution) if the behavior of their components are well defined and non-interacting. Eperience with programming parallel computers shows that it is extremely difficult to craft correct programs with interacting MIMD streams of control. Modification of such programs is even more difficult. In contrast, programming and debugging programs on SIMD machines, like Thinking Machines' Connection Machine, is very straight forward and much like serial programming. The high level center has a similar problem in controlling the parallel hardware of the rest of the brain. Because modifiability is so critical, I am not suprised a single stream of control was adopted there as well.

How much parallelism do we need in computers? I have argued that memory (either built in or aquired) is more important than computation. In brains neurons appear to be both the locus for memory and for computation. Having a large memory thus gives the system the opportunity to do massively parallel computation at no additional hardware cost. We may ask how much of this parallelism is useful in computers designed to perform functions similar to the brain's. The critical question in analysing the parallelism of the brain is how many of the neurons in the system are actually doing useful computation at any moment. Because of the direct coding scheme in the periphery, most of the computational power of the neural hardware there is wasted. For example, the neurons in a reflex arc only do useful computation for the organism when that reflex is instantiated. Similarly, there are a large number of internal pain receptors which fortunate people never know exist.

As we move to higher levels centers, it is important that neural hardware be shared amongst processes, both to enhance hardware utilization and to enhance generalization during learning. These two facets of distributed representations may be seen even in representing a single real parameter. In the presence of noise, a "coarse coded" representation in which the parameter's value space is decomposed into overlapping receptive fields gives much finer resolution than a nonover-lapping representation. Similarly, in learning a nonlinear mapping from one such domain to another one, the coarse coded representation automatically generalizes by continuity while the non-overlapping representation gives no generalization.

Clever algorithms can replace much brute parallelism. Exactly how much gain in computation can be achieved by this kind of shared representation is an important open question. I have argued [9] that many of the tasks performed by current neural network models can be performed far more efficiently on serial computers by using algorithms drawn from computational geometry. For example, I showed that a million item associative memory could be implemented over a billion times more efficiently using these algorithms rather than a simple neural network algorithm. This huge factor arises because much of the computational effort in a neural network simulation is unneccessary for the achievement of the goal. For example in a network of perceptron-like threshold units, each neuron corresponds to a hyperplane in the input space. Whether the neuron fires or not indicates which side of this hyperplane the input point lies on. A network represents a subset of states by approximating it by these hyperplanes. To determine if an input state is in the stored subset, it is compared with each hyperplane. In computer science we often do this kind of search by a divide and conquer technique. After we have tested the input against several hyperplanes, we can prune away many of the tests. In this way we can implement associative memories with only a logarithmic number of tests as opposed to the linear number performed by straightforward network models. By appropriate indexing of data, much associative retrieval can be implemented using only the very limited parallelism used in the address logic of the memory chips in serial computers.

Summary. The essential speed of the brain comes from caching a large store of procedures and experiences. Fast access to this information is provided by several modules acting in parallel each of which operates in parallel over its modality. These modules are controlled in serial by a higher level system which utilizes cached procedures built up by learning from slow declarative representations. The operation is serial because it is too difficult to plan and modify multiple streams of control. The associative memory functions for these operations may be implemented efficiently on serial machines by using clever organization principles. There appears to be ample opportunity to emulate parts of the brain's parallelism but we must be careful not to miss algorithms which make better use of engineering hardware.

References

[1] Korbinian Brodmann, *Vergleichende Lokalisationslehre der Grosshirnrinde in ihren Prinzipien dargestellt auf Grund des Zellenbaues*, (Barth, Leipzig, 1909).

[2] F. H. C. Crick and C. Asanuma, "Certain aspects of the anatomy and physiology of the cerebral cortex," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 2: Psychological and Biological Models*, edited by J. L. McClelland & D. E. Rumelhart, (MIT Press, 1986).

[3] David C. Van Essen and John H. R. Maunsell, "Hierarchical organization and functional streams in the visual cortex," *Trends in Neuroscience* (1983) 370–375.

[4] J. A. Feldman and D. H. Ballard, "Connectionist Models and Their Properties," *Cognitive Science*, **6** (1982) 205–254.

[5] Eric R. Kandel and James H. Schwartz, *Principles of Neural Science, Second Edition*, (Elsevier Science Publishing Co., Inc., 1985).

[6] Michael M. Merzenich and Jon H. Kaas, "Principles of Organization of Sensory-Perceptual Systems in Mammals," *Progress in Psychobiology and Physiological Psychology*, **9**, 1–42.

[7] Robert Neches, "Learning through Incremental Refinement of Procedures", in *Production System Models of Learning and Development* ed. Klahr *et. al.*, MIT Press (1987) 163–220.

[8] Paul Rosenbloom and Allen Newell, "Learning by Chunking: A Production System Model of Practice", in *Production System Models of Learning and Development* ed. Klahr et. *al.*, MIT Press (1987) 221–286.

[9] Stephen M. Omohundro, "Efficient Algorithms with Neural Network Behavior", *Complex Systems*, 1 (1987) 273–347.

[10] Shimon Ullman, "Visual Routines", in Pinker, *Visual Cognition*, MIT Press (1985) pp. 97–161.

TABLE OF CONTENTS

1988 SPRING SYMPOSIUM SERIES:

PARALLEL MODELS OF INTELLIGENCE: HOW CAN SLOW COMPONENTS THINK SO FAST?