A Reply to “Parallel Computation and the Mind-Body Problem”

Published as Krellenstein, M. (1987). A reply to "Parallel Computation and the Mind-Body Problem", *Cognitive Science*, *11*, 155-157.

Marc Krellenstein

New School for Social Research

Access Technology

The following is in reply to some points made in Paul Thagard’s “Parallel Computation and the Mind-Body Problem” (1986).

(1) Thagard argues that increased speed has metaphysical importance because only intelligent creatures (or machines) quick enough to adequately deal with the demands of their environment will survive. Thus, the best possible serial simulation of a parallel algorithm, while strictly possible, may be hopelessly slow. Lest we consider this conceptually irrelevant, Thagard admonishes us not to ignore such real-world limitations, observing that our theories relating matter and intelligence need no more account for the merely conceivable than Newtonian mechanics need explain worlds with negative gravitation.

But this still shows only that the hardware must be able to run the software fast enough to be useful/survive. A program running on a parallel machine that produced some sort of intelligence will also run on a serial machine, and this is enough to show the hardware irrelevant for explaining the nature, if not the evolution, of that particular intelligence. If the program runs too slowly on the serial machine to be useful we would not say that it no longer demonstrates intelligence but only that it is too slow, or that the particular approach, though successful, is impractical. (Some kind of practicality test is relevant to determining whether we have produced an intelligence that works in the same way as, as opposed to being functionally equivalent to, some aspect of human intelligence, but Thagard does not so constrain his position).

The analogy to negative gravitation is to instruct us not to muddy our thinking or burden any non-functionalist position with a purely theoretical multiple instantiation hypothesis and speculation about exotic serial machines. But the serial simulation of parallel processes and the portability of software are computational principles and everyday empirical realities, and they are deducible from, rather than premises of or motivation for, the computational paradigm embraced both by Thagard and the functionalists he attacks. These principles may prove less interesting if we find it impossible for intelligence to exist without massively parallel architectures very similar to the brain, but that will hardly make them less true or confer more than contingent importance on the particular hardware needed to achieve the requisite level of computational power.

(2) Thagard contrasts the functionalist's "sharp distinction between hardware and software" with the fuzzier separation of the two in current computers as evidenced by special-purpose computers with hard-wired software, or general-purpose computers with microprogrammed hardware. But functionalism is not concerned as much with different means of physical encoding as with the distinction between (virtual) machine and program, between interpretive mechanism and symbolic codes that are interpreted- for which "hardware" and "software" are a convenient shorthand. A given system may be viewed as consisting of multiple such virtual machines, and the point (if any) at which there is sufficient "hardness" as to render one essentially unmodifiable is rightly regarded as arbitrary.

At any given level, however, the line between virtual machine and program in a computer system is quite clear; it is what allows us to discuss the algorithms "followed" by a hard-wired chess machine, or to view the microprogramming level as part of the hardware virtual machine. What is important for the functionalist position is the equivalence between cognition and a program executed by some virtual machine.

(3) Thagard attempts to show that parallel hardware architectures offer not merely increased speed of processing but suggest qualitatively different programming approaches. One approach cited is being able to pursue multiple, sometimes improbable hypotheses simultaneously, causing Thagard to say that parallelism "lends itself to audacity."

But the position is overstated. A program running on a serial architecture limited to a set of heuristics for solving a problem might well adopt such an "audacious" approach given sufficient time to solve the problem relative to pursuing any one path. More likely, several paths might be pursued at different points in the process, foregoing the need and inefficiency (due to commonalities in approach) to pursue every approach from start to finish. Any of these paths might themselves represent improbable hypotheses if there is time to pursue all of the probable ones. This is a limitation shared by the parallel architecture: The ability to pursue the improbable is there only if there is sufficient processing power (sufficient processors) to pursue the probable.

Such an approach may in fact be more natural on a parallel architecture -- but probably because it is a simple answer to the difficult question of how to take advantage of multiple processors given an essentially serial approach to a problem for which there is not, or is not time for, a single algorithm certain of success. If such an approach turns out to be inadequate (perhaps because of number of likely and unlikely hypotheses) or if the goal is to harness the power of multiple processors to execute a complex algorithm guaranteed to produce the answer then there will be a need to more subtly distribute processing; and this is far more difficult.

Nor is it the case that pursuing multiple hypotheses is just a poor example of the extent of the differences suggested by parallel hardware. Rather, algorithms for parallel machines are often modified versions of (and more similar in essentials to than different from) serial algorithms, or directly exploit the presence of certain order-independent

steps in such algorithms, performing in parallel several steps that would otherwise be arbitrarily performed serially (see, for example, the discussion of algorithms for the radically parallel “connection machine” in Hillis, 1985). Conversely, traditional serial programs (e.g., compilers) are increasingly realized to contain sections that are perhaps most naturally implemented as parallel communicating tasks, perhaps in one of the several programming languages running on serial machines that support such concurrency. Such reformulations are only loosely (if at all) tied to the current or future existence of parallel architectures for running them.

References

Churchland, P. (1984). *Matter and consciousness*. Cambridge, MA: Bradford Books/MIT Press.

Hillis, D. (1985). *The connection machine*. Cambridge, MA: MIT Press.

Pylyshyn, Z. (1984). *Computation and cognition*. Cambridge, MA: Bradford Books/MIT Press.

Tanenbaum, A. (1976). *Structured computer organization*. Englewood Cliffs, NJ: Prentice-Hall.

Thagard, P. (1986). Parallel computation and the mind-body problem. *Cognitive Science, 10*, 301-318.