PROLEGOMENA TO ARTIFICIAL NEUROLOGY

Luís Moniz Pereira

CIUNL-13/81 November 1981

PROLEGOMENA TO ARTIFICIAL NEUROLOGY

Luís Moniz Pereira

Departamento de Informática
Universidade Nova de Lisboa
1899 Lisboa codex

Abstract

Potential bridging points between neurology and computer science are explored, with special emphasis on the hardware / software distinction.

Descriptive keywords: Artificial Intelligence, Neurology.

First, for some definitions.

That which is essentially the result of an intention is artificial, even if itself has intentionality – its intentions will be artificial. The intention behind which there is no other is natural.

For example, if the universe was intended by an all-powerful being who thereby realized its intentions, then all of it is artificial, including of course our own intentions. Another example: a computer built by a human being, and which manifests intentions, is wholly artificial, regardless of whether the human being, in turn, is artificial.

Granted, the usage of words changes. I can even foresee that the word natural, by virtue of the development of Artificial Intelligence, may come to be identically applied to many a computer, the originally artificial intentions of which will no longer be foreseeable by its human constructors. Then the creature forfeits independence from its creator.

This paper refers to an artificial world where there exist brains. I shall always thus omit the word artificial, and hence will write neuron rather than artificial neuron, intention rather than artificial intention, etc.

The artificial world alluded to, wherein there exist brains, was constructed by anonymous nondescript intentional beings, as an experimental laboratory for possibly confirming their conceptions about their own world. In particular, the brains therein are used as epistemological devices where the constructors of the world test, *in vitro*, some of their conceptions about their own cognitive processes, metaphysics, etc.

Such brains function according to principles previously tested by the constructors of the world on their computers, although the physiological and physical substrate of ones (the computers) and the others (the brains) differ substantially, the twain differing yet again from the material substrate supporting the mental activity of the world constructors.

Such state of affairs does not prevent ones be taken as models of the others. To the contrary, it is on that very principle of software independence of hardware, widely confirmed by their Computer Science, that the constructors of the world base their constructions. Consequently, when modeling their cognitive processes, they explore the potentialities of the model to improve, through a cognitive feedback loop, their own mental abilities.

Recourse to the nomenclature, paradigms, and techniques of computer science for brain modeling was not new to them. In fact, computer science had been, in the process, enriched with the nomenclature, paradigms, and techniques of the brain sciences.

Historically, that cross-fertilization began when it was acknowledged that only with the help of an instrument with the accumulated and organized complexity of a computing system (computer, peripherals, and programs) was it possible to deal in a rigorous fashion with the complexity of cerebral processes and structures of the constructors of the world and, subsequently, with those of the artificial brains they wished to build.

In particular, computer implemented models become well-defined, eminently observable in their formulation and their dynamics, and can be transformed incrementally in an expedite way. On the other hand, the observable dynamics of the models liberates neurology from its excessive emphasis on the pathology of lesions, and allows it to adopt a methodology more in line with the study of normal brain functioning. But pathology itself gains a new impetus with the possibility of simulating, in the model, a platter of specific and provoked lesions.

Let us recall, next, the admittedly most important principle originator of the introduction of the computer as a laboratory for brain models.

The principle of the distinction between software and hardware —of function and form, after all, in its simplest version, which in some way is present in any machine— appears finally clear-cut with the advent of the digital computer. The diversity of technologies employed to achieve the same function ever since the first computers, indeed confirms it. One same program is executable in physically different machines, precisely because at the level of discourse of the program the details of its execution, below an ascertainable level of analysis, are irrelevant, just as long the an identical level of discourse result is produced. In a crude analogy, one may say the ink colour and the handwriting are irrelevant to the message being conveyed.

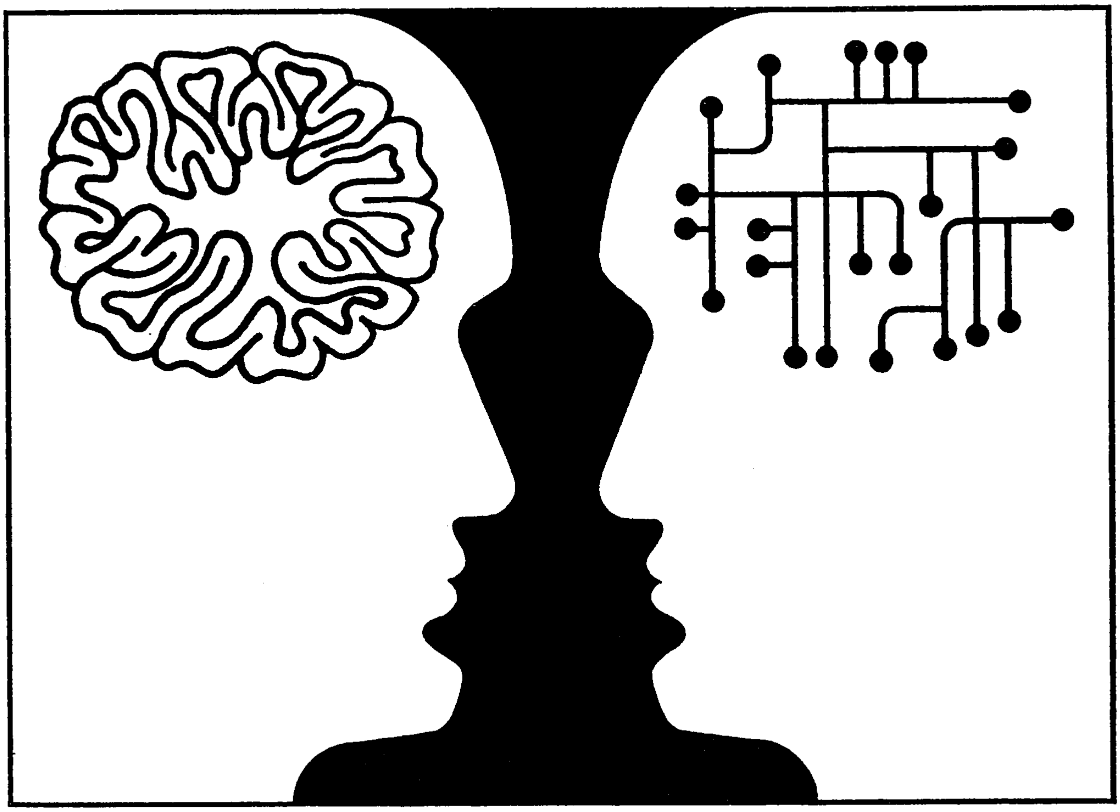
This distinction, let it be said, is susceptible of levels. That which is hardware is not necessarily things physical, but rather that which, at a certain level of analysis, is considered fixed, given, and whose analysis or non analysability is irrelevant for some purpose —e.g. RISC instructions, or glia cells' DNA.

Historically, in the first computers, that level coincided with the level of the physical parts of the machine —hence the confusion. Subsequently, that level moved in opposite directions, and its relativity became clear. On the one hand, the concept of abstract machine was introduced, that is, a given and non analysable fixed collection of instructions, mathematically defined, capable of supporting a set of software functions, independently of the physical processes and details that enact the implementation of the abstract machine on a physical one. On the other hand, the fixed physical components of one generation of computers have given place, in the next generation of computers, to partially programmable components, whose functions are software determinable (microprogramming). An old crude analogy would in this case be an IBM changeable ball typewriter. More to the point, some previously software defined functions, such as floating point arithmetic, became fully hardware available.

One main consequence, for artificial neurology, of this ever-present principle —whose precursors were the first axiomatic schools of thought— consists in the better focusing on the level of neurological analysis more appropriate for answering its questions.

Another main consequence has been, of course, the rising popularity of the computer among the neurologists as an instrument of simulation, given the advantage of being able to choose the level of abstraction of the simulation, including at the level of the neuron. In this task they helped themselves of the theory of black boxes of abstraction, successively developed by the computer scientists.

Many neurologists, indeed, had no difficulty in adjusting to the idea of neuron simulation because they recognized that their own conception of the neuron was already a rather abstract model of it, in contradistinction with the case of computer scientists, who can intimately know the computer's “circuits”, given these are built according to specifications.



The Epistemic Mirror

The software/hardware distinction is rich in consequences for artificial neurology. Namely it explains why the correspondence between function and its physical support is not compulsory. The physical hardware is not specific of any high level software function. Instead, it enables the execution of a variety of functions, in a distributed and non-localized way, exception being made for the hardware specific to the interface with peripheral organs, and to external information coding/decoding, as is the case of the nervous system, say, according to A. R. Luria. As the cerebral processes gain in abstraction and level of integration of several sensory sources, the neurons which support them become less specific and independent of the origin of the information. How else would integration be possible if it were not so?

It is a fact that beyond the interface with the exterior there exist specific hardware which is also specialized. But such specificity could conceivably be realized in other ways, *pace* organic chemistry, so that no specific software actually requires a specific hardware.

Another consequence of the hardware/software distinction concerns the notion of appropriate explanation level. A program can be understood, in its function or dysfunction, in terms of itself, of its own level of discourse. Of course, its dysfunction can originate in a dysfunction of the underlying hardware, but in that case it manifests itself in a bizarre behaviour, not comprehensible at the program's level of discourse, and not specific to that program. Complementarily, its function can be described resorting to the hardware level, but such description does not constitute an appropriate level of explanation because, being too detailed, it is not generalisable. The following analogy, adapted from Putnam, is an example of what I mean.

Imagine a rigid panel, with a circular hole of 10 cm in diameter, and a square hole of 10 cm side. One wants to explain why a rigid cube, of 9 cm side, goes through one of the holes but not the other. The adequate level of explanation resorts to the geometrical concepts and principles involved. A possible but in appropriate level of an explanation would consider the quantum-physical properties of the materials present, say glass for the panel and aluminum for the cube, and explain the impossibility of passage of the cube through one the circles in terms of mechanical resistance, for any approaching trajectory.

Such an explanation, because overly specific, is generalisable with difficulty to other constituent materials, say iron and granite. There is a level of analysis below which the explanation loses in generality and becomes unnecessary, as long as all the relevant properties are guaranteed to be fixed at a that level —in the example given, the form invariance of the elements in presence, relative to any trajectory followed and/or their physical composition.

Finally, in a computer, the software prevails, in general, over the hardware. Though the hardware supports and causes the execution of the software, the initiative belongs, more times than not, to the software. It is the software that chooses and provokes the coming into activity of the appropriate hardware at each step. Such activity consists in consulting the instructions stored in memory, and in executing such software instructions in the hardware, with the result that instruction selected hardware is provoked into activity, closing the circle. In this way, the teleology of the software is kept in charge, notwithstanding the underlying causality of the physical hardware.