on the issue, my contention is that it's less than likely that they would make any difference. The claim that the brain is a specific kind of an information-processing mechanism, and that information-processing is necessary (even if not sufficient) for cognition, is non-trivial and generally accepted in cognitive (neuro)science. I will not develop the positive view here, however, as it was already stated sufficiently clearly to my tastes in book-length accounts.[1] Instead, I will go through the objections, and show that they all fail just because they make computationalism a straw man.

SOFTWARE AND NUMBER CRUNCHING

One fairly popular objection against computationalism is that there is no simple way to understand the notions of *software* and *hardware* as applied to biological brains. But the software/hardware distinction, popular as the slogan "the mind to the brain is like the software to hardware,"[2] need not be applicable to brains at all for computationalism to be true. There are computers that are not program-controllable: they do not load programs from external memory to internal memory to execute them. The most mundane example of such a computer is a logical gate whose operation corresponds to a logical connective, e.g., disjunction or conjunction. In other words, while it may be interesting to inquire whether there is software in the brain, there may as well be none, and computationalism could still be true. Hence, the objection fails, even if it is repeatedly cited in popular press.

Another intuitive objection, already stated (and defeated) in the 1950s, is that brains are not engaged in number-crunching, while computers, well, compute over numbers. But if this is all computers do, then they don't control missiles, send documents to printers, or display pictures on computer monitors. After all, printing is not *just* number crunching. The objection rests therefore on a mistaken assumption that computers can only compute numerical functions. Computer functions can be defined not only on integer numbers but also on arbitrary symbols,[3] and as physical mechanisms, computers can also control other physical processes.

SYMBOLS AND MEANING

The notion of a symbol is sometimes interpreted to say that symbols in computers are, in some sense, abstract and formal, which would make computers strangely dis-embodied.[4] In other words, the opponents of computationalism claim that it implies some kind of dualism.[5] However, computers are physical mechanisms, and they can be broken, put on fire, and thrown out of the window. These things may be difficult to accomplish with a collection of abstract entities; the last time I tried, I was caught red-handed while committing a simple category mistake. Surely enough, computers are not *just* symbol-manipulators. They do things, and some of the things computers do are not computational. In this sense, computers are physically embodied, not unlike mammal brains. It is, however, a completely different matter whether the symbols in computers mean anything.

One of the most powerful objections formulated against the possibility of Artificial Intelligence is associated with John Searle's Chinese Room thought experiment.[6] Searle claimed to show that running of a computer program is not sufficient for semantic properties to arise, and this was in clear contradiction to what was advanced by proponents of Artificial Intelligence who assumed that it was sufficient to simulate the syntactic structure of representations for the semantic properties to appear; as John Haugeland quipped: "if you take care of syntax, the semantics will take care of itself."[7] But Searle replied: one can easily imagine a person with a special set of instructions in English who could manipulate Chinese symbols and answer questions in Chinese without understanding it at all. Hence, understanding is not reducible to syntactic manipulation. While the discussion around this thought experiment is hardly conclusive,[8] the problem was soon reformulated by Stevan Harnad as "symbol grounding problem":[9] How can symbols in computational machines mean anything?

If symbol grounding problem makes any sense, then one cannot simply assume that symbols in computers mean something just by being parts of computers, or at least they cannot mean anything *outside* the computer so easily (even if they contain instructional information[10]). This is an assumption made also by proponents of causal-mechanistic analyses of physical computation: representational properties are not assumed to necessarily exist in physical computational mechanisms.[11] So, even if Searle is right and there is no semantics in computers, the brain might still be a computer, as computers need no semantics to be computers. Maybe something additional to computation is required for semantics.

Let us make the record straight here. There *is* an important connection between the computational theory of mind and the representational account of cognition: they are more attractive when both are embraced. Cognitive science frequently explains cognitive phenomena by referring to semantic properties of mechanisms capable of information-processing.[12] Brains are assumed to model reality, and these models can be computed over. While this seems plausible to many, it's important to remember than one can remain computationalist without assuming representationalism, or the claim that cognition requires cognitive representation. At the same time, a plausible account of cognitive representation cannot be couched merely in computational terms as long as one assumes that the symbol grounding problem makes sense at least for some computers. To make the account plausible, most theorists appeal to notions of teleological function and semantic information,[13] which are not technical terms of computability theory nor can be reduced to such. So, computers need something special to operate on inherently meaningful symbols.

What made computationalism so strongly connected to cognitive representations was the fact that it offered a solution to the problem of what makes meaning causally relevant. Many theorists claim that just because the syntax in computer programs is causally relevant (or efficacious), so is the meaning. While the wholesale reduction of meaning to syntax is implausible, the computational theory of mind makes it clear that the answer to the question includes the causal role of the syntax of computational vehicles. Still, it is not an objection to computationalism itself that it does