Algorithms Implemented in Space and Time

Paul Schweizer[1]

Abstract. A fundamental question regarding computation viewed as a physical phenomenon concerns the criteria under which a physical system can properly be said to implement an abstract formal procedure. I advocate the Simple Mapping Account (SMA) and defend this position against a number of critiques, including semantic and causal accounts. In line with SMA I support the conclusion that realizing or implementing an abstract computational procedure is not an intrinsic property of physical systems, but rather is based on a purely observer-dependent act of ascription. This 'anti-realist' conclusion has traditionally been invoked in an attempt to undermine the Computational Theory of Mind (CTM), which in turn has led supporters of CTM to criticize SMA and propose competing accounts. In contrast, I argue that the version of CTM directly threatened by SMA is one that should not be accepted in any case, and propose an alternative strategy for those who would defend CTM against charges of 'triviality'.

1 INTRODUCTION

A fundamental question regarding computation viewed as a physical phenomenon concerns the criteria under which a physical system can properly be said to implement an abstract formal procedure. A very straightforward and elegant account articulated by Putnam [1] is based on a simple mapping between formalism and physical structure. Accordingly, a physical system *P* performs a computation *C* just in case there is a mapping from the physical states of *P* to the abstract computational states of *C*, such that the transitions between physical states reflect the abstract state transitions as specified by the mapping. The minimalism and elegance of the Simple Mapping Account (SMA) make it the natural choice as the in-principle standard for physical implementation − it takes the Mathematical Theory of Computation (MTC) as its starting point and adds no substantive assumptions.

Central to MTC is the intuitive notion of an effective or ‘mechanical’ procedure, which is simply a finite set of instructions for syntactic manipulations that can be followed by a machine, or by a human being who is capable of carrying out only very elementary operations on symbols. A key constraint is that the machine or the human can follow the rules without knowing what the symbols *mean*. The notion of an effective procedure is obviously quite general – it doesn’t specify what form the instructions should take, what the manipulated symbols should look like, nor precisely what manipulations are involved. The underlying restriction is simply that they are finitary and can proceed ‘mindlessly’ i.e. without any additional interpretation or understanding. So there are any number of different possible frameworks for filling in the details and making the notion rigorous and precise. Turing’s ‘automatic computing machines’ [2] (TMs), supply a very intuitive and elegant rendition of the notion of an effective procedure, and in the ensuing discussion TMs will be taken as the conceptual archetype. But there is a variety of well known alternative frameworks, including Church’s Lambda Calculus, Gödel’s Recursive Function Theory, Lambek’s Infinite Abacus Machines, etc.

Turing machines and other types of computational formalisms are *mathematical abstractions* and don’t exist in real time or space. In order to perform *actual* computations, an abstract Turing machine must be realized by a suitable arrangement of matter and energy, and as Turing observed long ago [3], there is no privileged or unique way to do this. Like other abstract structures, Turing machines are *multiply realizable* - what unites different types of physical implementation of the same abstract TM is nothing that they have in common as physical systems, but rather a structural isomorphism expressed in terms of a higher level of description. Hence it’s possible to implement the very same computational formalism using modern electronic circuitry, a human being executing the instructions by hand with paper and pencil, a Victorian system of gears and levers, as well as more atypical arrangements of matter and energy including beer cans serving as tokens of the symbol ‘1’ and rolls of toilet paper serving as the tape.

Adopting notational conventions introduced in Schweizer [4], let us call this ‘downward’ multiple realizability, wherein, for any given abstract structure or formal procedure, this *same* abstract structure can be implemented via an arbitrarily large number of *distinct* physical systems. And let us denote this type of downward multiple realizability as ‘↓MR’. After the essential foundations of MTC were laid, the vital issue then became one of engineering – how best to utilize state of the art technology to construct rapid and powerful physical implementations of our abstract mathematical blueprints, and hence perform actual high speed computations *automatically*. This is a clear and deliberate ↓MR endeavour, involving the intentional construction of artefacts, painstakingly designed to follow the algorithms that we have created. From this top-down perspective, there is an obvious and pragmatically indispensible sense in which the hardware that we have designed and built can be said to perform genuine computations in physical space-time.

2 COMPUTATIONAL THEORY OF MIND

According to the widely embraced computational theory of mind (CTM), which underpins cognitive science, Strong AI and various allied positions in the philosophy of mind, computation (of one sort or another) is held to provide the scientific key to explaining and, in principle, reproducing mentality artificially. The paradigm maintains that cognitive processes are essentially computational processes, and hence that intelligence in the

[1] Institute for Language, Cognition and Computation, School of Informatics, Univ. of Edinburgh, EH8 9AD, UK. Email: **paul@inf.ed.ac.uk**.

natural world arises when a material system implements the appropriate kind of computational formalism. Various critics of CTM have put forward a family of 'trivialization arguments', stemming directly the SMA above. The arguments are based on the contention that the notion of a physical system implementing a computational formalism is overly liberal to the point of vacuity, since a mapping will obtain between any sufficiently complex physical system and virtually any computational formalism. This would appear to trivialize CTM, since whatever computational formalism is held to account for our cognitive attributes will also be realized by a myriad of other 'deviant' arrangements of matter and energy, from buckets of water to microwave ovens to possibly even stones. By CTM it would seem to follow that such obviously insentient systems have the same cognitive attributes that we do, since they can be interpreted as implementing exactly the same computations. For example, assuming SMA, Putnam offers a proof of the thesis that *every* open physical system can be interpreted as the realization of *every* finite state automaton. In a closely related vein, Searle [5] argues that virtually any physical system can be interpreted as following virtually any program. Thus hurricanes, our digestive system, the motion of the planets, even an apparently inert lecture stand, all possess a level of description at which they instantiate any number of different abstract formal procedures. The stomach has inputs, internal processing states and outputs, and if one wanted to, one could interpret the inputs and outputs as code for any number of different symbolic processes. And in [6] Searle attempts to illustrate the extreme conceptual looseness of the notion of implementing an abstract formalism by claiming that the molecules in his wall could be interpreted as running the WordStar program.

In this manner, critics of CTM utilize SMA to argue for 'multiple realization' in the form of a one-to-many-relation between physical structure and abstract interpretation. Again, adopting notational conventions introduced in Schweizer, let us label multiple realizability in this direction, wherein any given *physical system* can be interpreted as implementing an arbitrarily large number of different *computational formalisms* 'upward MR' and denote it as '↑MR'. The basic import of ↑MR is the *non-uniqueness* of computational ascriptions to particular physical systems. In the extreme versions suggested by Putnam, Searle, and more recently Bishop [7], there are apparently no significant constraints whatever – it is possible in principle to interpret every open physical system as realizing every computational procedure. Let us call this extreme version 'universal upward MR' and denote it as '↑MR*'. Mere ↑MR is weaker than ↑MR*, since the former does not assert that there are no salient constraints, and hence ↑MR would be consistent with the denial that, e.g., the molecules in Searle's wall can in fact be interpreted as implementing the WordStar program, although every physical system is still interpretable as implementing some very large set of distinct computations.

In the present discussion I will not argue for or against ↑MR* but restrict consideration to the more modest ↑MR. In view of ↑MR, it's still never the case that any given computational interpretation of a physical system is privileged or unique, and this is far more difficult to deny than the powerful and broad sweeping ↑MR*. In turn, the non-intrinsic status of computation would seem to follow as a direct consequence of mere ↑MR alone. As long as there are at least two distinct interpretations, there is no objective fact of the matter regarding *which* computation is 'actually' being performed, nor which of the alternatives is the 'correct' or 'real' account. And this is because the computation itself is not an intrinsic property of the physical device, and is instead dependent on a human observer to supply the various alternative interpretations.

This is not to say that it's purely a matter of caprice, and that there are no objective constraints that the interpretation must satisfy. Instead, the situation is perhaps comparable to the distinction between natural kinds, such as water, and conventional kinds, such as being a table. Even though membership in either kind might be based on criteria whose satisfaction (or not) is a matter of objective truth, still the criteria for conventional kinds are not intrinsic, and there is nothing about the particular arrangement of matter now holding up my desk top computer which makes it intrinsically a table. The salient criteria stem purely from human practices and stipulations rather than from, e.g., fundamental microstructure or natural law.

3 THE SEMANTIC ACCOUNT

Advocates of CTM typically attempt to defend their paradigm against such trivialization arguments by rejecting SMA as *itself* overly liberal, and advocating additional constraints on the notion of 'true' or 'genuine' implementations, to distinguish them from the many presumably 'false' cases countenanced by SMA. In this manner, the hope is that the myriad of apparent counterexamples generated by SMA will be screened off as 'fakes', and the cogency of CTM thereby preserved. Three primary categories of constraint put forward by defenders of CTM include the semantic account (SA), the causal account and the counterfactual account. Each of these will be explored and critiqued in turn, beginning with the first.

Concerning the semantic account of computation, the 'received view' in the philosophy of mind is that computation must involve representational content. This view is encapsulated in Fodor's [8] famous edict that there is "no computation without representation". According to SA, computation is stipulated to be the processing of *representations*, and only physical states that are 'representational' can serve as implementations of the computational states in question. However, I will presently argue that this move is infelicitous for a number of reasons, and later in the paper will propose an alternative strategy for those who would defend CTM against trivialization. The SA is infelicitous because:

(1) It advocates a departure from MTC, whereas MTC is the canonical source of our conceptual grasp of computation. As above, classical computation is defined as rule governed symbol manipulation, and a key proviso is that the rules can be followed without any knowledge of what the 'symbols' are supposed to mean. As Piccinini [9] aptly observes, representational content plays no role whatever in MTC.

(2) MTC is clear and rigorous, while the further restrictive notion of 'representation/reference' invoked by SA is imprecise and problematic. Hence this is a retrograde step from clarity and generality to narrowness and potential obscurity. Indeed, given the notorious difficulties in providing a satisfactory rendition of 'representation' in objective scientific terms, SA is in the rather ironic position of promulgating a restriction on the global notion of computation in the physical world that is itself unlikely to be successfully naturalized.

(3) Our computational artefacts are the paradigmatic instances of physical computation and can yield any number of counterexamples to SA. A Turing machine designed to compute the values of a particular truth function, say inclusive disjunction, can be easily *reinterpreted* as computing conjunction instead, simply by flipping the intended reference of the symbols '0' and '1'. There is no independent fact of the matter regarding what these syntactic tokens 'really represent' − their referential value is entirely dependent upon an arbitrary scheme of interpretation. As a consequence, there is no unique meaning determined by the formal procedure as such, and a multitude of distinct and *incompatible* interpretations are always possible. This highlights a fundamental flaw related to (1) above: computation is essentially pure *syntax* manipulation, and how the syntax is interpreted is an *additional* feature not intrinsic to computation *per se*. SA stipulates that this extrinsic feature is essential, even though the discipline of Computer Science makes no such claim. I would argue that SA commits the mistake of conflating 'computation' *simpliciter* with 'syntax manipulation under an intended interpretation'. But of course, the formal syntax manipulation can take place in the absence of *any* interpretation. Hence in a very clear, rigorous and universal sense, contra Fodor there *is* computation without representation, because semantics is purely extrinsic to effective procedures as such.

(4) The primary reason for making the foregoing conflation and attempting to tether the notion of computation to some story about representation does not stem from any issues concerning the general theory of computation itself, but rather is driven by a particular stance within a specialized explanatory project in the *philosophy of mind*. And this is an overly parochial source for deriving restrictions on physical computation in general.

In response to these infelicities, I would contend that SA is not a viable approach to computation *per se*. So rather than adopt this limiting and undermotivated standpoint, we should instead take MTC, one of the towering intellectual achievements of 20th century theorizing, as the canonical framework for understanding computation in the general abstract sense, and we should adopt SMA as the global, theory-neutral template for the concomitant notion of physical implementation. These two standards are utterly rigorous, comprehensive and impartial, and are not themselves in any need of tweaks or alterations. As noted above, the main reason for the restrictions imposed by SA stems not from the general nature of computation, but instead serve to protect vested theoretical interests held by *other* disciplines that assume computation as a primary ingredient in their specialized explanatory frameworks. In section 7. I will return to the issue of CTM and propose an alternate strategy for those who would defend the paradigm against charges of empirical vacuity.

4 CAUSAL CONSTAINTS

In response to ↑MR* and the associated trivialization arguments, a number of other authors including Chrisley [10], Chalmers [11], Copeland [12], and Block [13] propose further constraints on computational interpretations. Two of the most intuitively compelling restrictions are supplied by (i) causal and (ii) counterfactual considerations. Although both (i) and (ii) are plausible and natural suggestions, I will argue that neither are ultimately unsuccessful in blocking ↑MR*.

Regarding point (i), Chalmers, for example, contends that it is a necessary condition (for counting as a legitimate implementation) that the pattern of abstract state transitions constituting a particular run of the computational procedure on a particular input, must map to an appropriate transition of physical states of the machine, where the relation between succeeding states in this sequence is governed by proper causal regularities. This suggestion constitutes quite a natural and immediate corrective measure in response to the extreme laxity that might seem to underwrite ↑MR*, since the physical states in the chronological progression exploited by Putnam's method have no nomological connection.

Nevertheless, I would argue that the constraint is too strong in general and rules out cases which should not be excluded. There are many instances of sequences of physical states that we count as realizing a particular computation simply because, according to our abstract blueprint, the *correct series* of physical sate transitions *actually occurs*. For example, standard computers rely on a hierarchy of levels of description pertaining to 'virtual machines', and it is entirely natural to construe high level virtual machines as genuinely implementing computations, even though the states at this level of description are not themselves causally connected. Furthermore, we do not need to know *anything* about the complex underlying architecture nor its causal underpinning in the electromechanical hardware, in order to ascertain that the respective computation is successfully being carried out. All we need to take into account is what actually happens at the given level of virtual machine description.

In an analogous manner, consider the following sequence of Turing Machine tape configurations, where each digit corresponds to the contents of one square of the tape, and the underlined digit to the currently scanned square:

Begin $\underline{1}10100$
$1\underline{1}0100$
$11\underline{0}100$
$110\underline{1}00$
$1101\underline{0}0$
$110\underline{1}00$
$110\underline{0}00$
$11\underline{0}000$
$11\underline{1}000$
$1\underline{1}1000$
$\underline{1}11000$ Halt

This sequence is the implementation of a particular program for addition positive integers in monadic notation, and constitutes a computation of 2+1=3. Yet the entries in this sequence bear no decipherable *causal* relations to each other, and now that the sequence is completed it can be revisited at any future date and still confirmed as a computation of 2+1=3, even though there is no longer *any* causal or even temporal connection between the already finished entries in the sequence of digits constituting the implementation.

Similarly, in various situations where a human being is following an abstract computational procedure, the transition from one state to the next is not causal in any straightforward physical or mechanical sense. When I take a machine table set of instructions specifying a particular TM and then perform a given computation with pencil and paper by sketching the configuration of the tape at each step in the computation, the transitions sketched on the piece of paper are *not themselves* causally connected: one sketch in the sequence in no way causes the next. It is only through my understanding and intentional choice to execute the procedure that the next state appears on the paper. Clear-cut physical causation of the sort required by Chalmers comes in only very indirectly, as in light rays illuminating the page and allowing me to see the symbols, and at an elementary and extraneous level, as in the friction between the pencil lead and the paper's surface causing various marks to appear.

Yet this is a perfectly legitimate and indeed paradigmatic case of implementing a Turing machine. And likewise in the Chinese room, it is merely through Searle's *understanding* of English, his voluntary choice to behave in a certain manner, and a number of highly disjointed physical processes (finding bits of paper in a certain location, turning the pages in the instruction manual, all mediated by the human agent) that the implementation takes place. Searle, as an intentional agent, is choosing to cause various things to happen in accordance with a set of rules that he chooses to follow. And Searle's intentionally characterized behaviour is not something that we currently have any hope of ever being able to recast in terms of causal regularities at the purely *physical level of description*.

One might rejoin that, at least in principle, it's still theoretically *possible* to characterize the overall system purely in terms of natural laws and causal regularities, *a la* Dennett's [14] Martian superscientist, who doesn't require the intentional stance to predict human behavior. And while this may well be true in principle, I don't think it really helps, since *we* can't do so, and we're the ones interpreting Searle as performing a computation. One could perhaps simply assert that, since Searle is indeed performing a computation, then there *must be* the appropriate sort of causal regularities underpinning his behavior, even though we don't know what they are and can't foresee a time in the future when we will. But why must there be such regularities? – presumably because Searle is performing a computation and the causal account is true... But such a line of reasoning would clearly beg the question. Undoubtedly Searle's behavior must have a *cause*, but from this it does not follow that it is governed by any physically characterizable regularities that even remotely resemble the structure of the algorithm.

Furthermore, we can let chance and randomness into the scenario. Suppose at each step in the computation Searle flips a coin, and will only follow the rule if the coin comes up heads. And suppose further that, for a particular run on an input question, the coin comes up heads every time and Searle successfully outputs the answer. He has still implemented the formalism, even though this outcome was not predictable on the basis of causal regularities or natural law.

And how could we know that the right causal connections are preserved via Searle's agency, even in the cases where he *sincerely intends* to follow the rule book? – how do we know that at some crucial stage he did not *misunderstand* the rules, and the step he actually intended to perform would have been a mistake, but that by a slip of attention he did *not* perform the step he intended but rather accidentally performed the correct one? As long as the step was correct we should count this as a physical realization of the abstract procedure. And indeed, how do we know that such self cancelling pairs of mistakes don't sometimes occur in our computational artefacts?

In such cases, the physical sequences count as implementations simply because what can be interpreted as the appropriate states in the procedure *occur* in the correct linear order. In other words, the intended mapping, *a la* SMA, has been preserved. And this highlights a very key point – the fundamental criterion is *normative* rather than causal. Underlying causal considerations are the wrong level of analysis, partly because there is then no sense in which error or malfunction can occur. Physical processes 'obey' natural law-like regularities in a purely descriptive manner, and over the time evolution of a physical system the trajectory of states in the process may or may not correspond to our projected computational interpretation. If not, then there has been a 'malfunction' in the hardware. But of course, systems governed by causal regularities cannot malfunction as such, and it is only at a higher and *non-intrinsic* level of description that 'breakdowns' can take place. We characterize these phenomena as hardware malfunctions, not because underlying scientific laws have been broken, but rather because the intended interpretation, which is prescriptive and non-intrinsic in nature, has. And there is *always* a non-zero probability of error for any algorithms executed in physical space-time. Files become 'corrupted', signal transmissions convey 'misinformation', overheating induces processing 'faults', etc.

All these mechanically mundane occurrences take place in complete accord with the causal regularities that govern the evolution of physical systems through time. Hence their status as 'malfunctions' has nothing to do with causal considerations, and they can be interpreted as such *only relative to* our projected formal mapping. In such cases, the physical system *fails* to count as an implementation on the purely normative grounds that the correct sequence of states did not occur, and so our intended mapping is violated. To be sure, there will be an underlying causal story for why the hardware performed the way it did, but this has nothing to do with the question of whether or not the device has successfully implemented the algorithm in question. Likewise, there will be a causal story for why the hardware performed the way it did when our projected interpretation is *respected* and the physical device counts as a 'valid' implementation. In both cases the issue of success or failure is determined relative to our intended interpretation, and hence is settled on purely normative rather than causal grounds. And this is in perfect agreement with SMA.

Questions regarding the mechanics of *how* the correct sequence of states happen to occur are not relevant to answering the question of whether or not the procedure has been physically implemented. In the Chinese Room we can know that the procedure has been implemented without knowing how Searle himself (or his brain) manages to do the requisite internal processing and control his limbs in order to make the correct marks on the slips of paper. The physical how is a *different* question, and is not on the same level of analysis as that invoked when determining whether or not the desired computation has been performed. But this then critically loosens the requirements for counting a physical system as instantiating a program. As long as what can be described or interpreted as the correct sequence of states actually occurs, then the underlying mechanics of how this takes place are not strictly relevant.

The right sort of causal connections and regularities are needed if the instantiation in question is to be *fully automatic*, and if we want to be able to rely on the automatic device to perform systematically correct computations yielding outputs with the potential to supply us with new information. And although this is the engineering norm when constructing and interpreting computational artefacts, it does not exhaust the general space of possibilities. The causal requirements advocated by Chalmers constitute (at best) a sufficient but not a necessary condition – in the general case we must still allow for chance and human agency to play a role, as well as chronological sequences of states that are not themselves governed by overarching causal regularities.

5 COUNTERFACTUAL CONSTAINTS

In line with (ii) above, Chalmers' proposed *counterfactual* requirement is aimed at another apparently 'slack' feature incorporated by Putnam and the SMA, *viz*. the mapping from formalism to physical system is defined for only a single run, and says nothing about what *would* have happened *if* a different input had been given. And it is objected that this is too weak to satisfy the more rigorous operational notion of being a 'genuine' realization. However, in response to Chalmers' (again quite natural) proposal, it is worth noting that for a physical system to realize a rich computational formalism with proper input and output capacities, such as an abstract TM, this will always be a matter of *mere approximation*. For example, any given physical device will have a finite upper bound on the size of input strings it is able to process, its storage capacities will likewise be severely limited, and so will its actual running time. In principle there are computations that formal TMs can perform which, even given the fastest and most powerful physical devices we could imagine, would take longer than the lifespan of our galaxy to execute. Hence even the fastest and most powerful physical devices we could envision will still fail to support all the salient counterfactuals.

It will never be possible to construct a complete physical realization of an abstract TM – the extent to which the concrete device can execute the full range of state transitions of which the abstract machine is capable will always be a matter of *degree*. For example, consider the sequence of configurations exhibited in section 4, which constitute an implementation of a particular TM program for addition of positive integers expressed in monadic notation. The extremely simple program can be specified in terms of the following set of six quadruples

$$\mathbf{q_1 1 B q_{1;}\ q_1 B R q_{2;}\ q_2 1 R q_{2;}\ q_2 B 1 q_{3;}\ q_3 1 L q_{3;}\ q_3 B R q_4}$$

where the first element in each quadruple (e.g. $\mathbf{q_1}$ in the first quadruple) is the current state, the second element is the currently scanned symbol (either **1** or **B** for blank, i.e. 0) the third element is the overt action (*move* **R** or **L** one square, or *print* a **1** or a **B**), and the last element is the covert 'act' of entering the next state. The exhibited sequence of configurations depicts the behavior of the machine given 2 and 1 as inputs, and it's a simple matter to implement *this particular* computation in space and time. However, there is no finite upper bound on the size of input strings that this abstract machine can deal with. The set of six quadruples yields a mathematically well defined and effective procedure for adding two strings, each of which contains in excess of, say, $10^{1000000000000000000000}$ 1's. And although it may be a simple matter to construct an implementation of the machine capable of carrying out computations on small input numbers, it's not physically possible for any such implementation to carry out the computation for the astronomical inputs above. Hence *no* physical implementation of this simple four state TM can deal with the full range of *possible* inputs.

So, in general, the class of counterfactual cases on alternative inputs with which a physical realization can cope is by necessity limited – not all counterfactual cases will be supported by *any* physical device implementing any TM. And this renders the appeal to counterfactuals inescapably *ad hoc*. The restrictive strategy demands that the mapping be able to support counterfactual sequences of transitions on inputs not actually given - but precisely *how many* inputs not actually given? One, two, twenty million? For any implementation, there will be a finite upper bound on the size of input string it can process, and beyond that size there will be *infinitely many* potential inputs for it will not be able to perform the salient computation.

This indicates that there is no clear or principled cut off point demarking 'genuine' implementations from 'false' ones in terms of counterfactuals. As another, more common place, illustration of the *ad hoc* nature of the appeal to counterfactuals, consider a standard pocket calculator that can intake numbers up to, say, 6 digits in decimal notion. Is this a 'false' realization of the corresponding algorithm for addition, since it can't calculate $10^6 + 10^6$? It's an *approximate* instantiation which is nonetheless exceedingly useful for everyday sums. It will always be a matter of degree how many counterfactuals can be supported, where a single run on one input is the minimal case. Where in principle can the line be drawn after that? It's a matter of our purposes and goals as interpreters and epistemic agents, and is not an objective question about the 'true' nature of the physical device as an implementation. In some cases we might only be interested in the answer for a single input, a single run.

Hence for a physical device to successfully 'perform a computation' is distinct from 'fully instantiating a computational formalism'**.** Performing a computation is an occurrent series of events, an actual sequence of physical state transitions yielding an output value in accord with the normative requirements of the mapping from abstract formalism to physical process. And this can be satisfied in the case of computing the value of a single output on a given input. In contrast, instantiating a complete computational formalism is a much more stringent and hypothetical notion, requiring appeal to counterfactuals, and as above, this will only ever obtain as a matter of degree. In light of this distinction, it is clearly possible for a physical device to successfully perform a computation *without* instantiating a complete computational formalism, which distinction in turn fatally undermines the theoretical force of counterfactuals in attempting to determine whether a physical process has 'really' performed a computation.

In this section I've argued that the question of whether a given physical process or device implements a computational formalism does not have a proper yes/no answer, and even in the most clear cut and paradigmatic case of a custom designed artefact, the implementation is a finitely bounded approximation which must fall far short of the abstract ideal. And the problem of error noted in the preceding section can also be seen to lend strong support to the claim that there is no realist true/false answer. Even in the case of a custom designed artefact executing a single run on one input, there is always a non-zero probability of error, which indicates that the physical device is merely a concrete 'estimate' of the abstract mathematical blueprint, and satisfies the normative procedures only as a matter of degree. The artefact may actually compute the correct value in the given case, but suppose we then start considering the counterfactual class of alternative inputs, and on one of these possible alternative runs it would have made an error and outputted an incorrect value. Surely this very genuine counterfactual possibility does not undermine the actual case, and support the claim that the machine does not 'really' implement the intended algorithm. But then neither does tactic (ii) successfully rule out ↑MR* (nor weaker but extremely wide ranging versions of ↑MR).

Furthermore, Bishop has importantly extended the SMA strategy to show that any predetermined finite set of counterfactuals *can* be accommodated on this approach. From this I would conclude that the underlying and more general constraint of concern to those who would delimit the range of physical implementation is neither causal nor counterfactual. Instead, the point to emphasize is that in ↑MR* exercises of this sort, the mapping is entirely *ex post facto*. The abstract procedural ‘trajectory’ is already known and used as the basis for interpreting various state transitions in the open system and hence characterizing it as an implementation. As long as this *ex post facto* tactic is allowed, then even finite sets of counterfactuals can be included. And as emphasized above, our actual computational artefacts are themselves only capable of handling finite sets of counterfactuals. Hence the pivotal issue is not counterfactuals but rather the *ex post facto* character of the mapping. I will return to this theme in a subsequent section of the paper.

6 SYNTAX, SEMANTICS, PHYSICS

At the abstract, formal level, computation is an essentially syntactic phenomenon, and how we choose to interpret arrangements of matter and energy as constituting, say, tokens of an abstract syntactic type, and thus specifying an implementation of the basic computational vocabulary, is entirely independent of physical composition. For example, in the downward ↓MR direction there is a more or less limitless diversity in the ways in which material patterns and arrangements can be viewed as implementing the binary notation of ‘0’ and ‘1’, from ink marks on a piece of paper, stones placed in wooden boxes, patterns on old-fashioned punch cards, electric voltages, beer cans positioned on rolls of toilet paper, … And this applies in the reverse ↑MR direction as well, wherein the same stones placed in wooden boxes can be interpreted as implementing any number of distinct computational formalisms.

Classical computation is rule-governed syntax manipulation, and it is no more intrinsic to physical configurations than is syntax itself. It is also worth observing that discrete states are themselves idealizations, since the physical processes that we interpret as performing computations are in fact continuous, and we must abstract away from the continuity of the underlying substrate and impose a scheme of conventional demarcations to attain discrete values. Hence even this elemental building block of digital procedures must be projected on to the natural order from the beginning. The irresistible conclusion to be drawn is that there is a fundamental gap separating ‘concrete’ physical reality from the human-based ascriptions of abstract syntactic features.

In turn, there is yet *another* fundamental gap separating abstract syntactic features from their semantic interpretation. Just as syntax is not intrinsic to physics, so too semantics is not intrinsic to syntax. Just as being an instance of the spoken English sentence ‘The cat is on the mat’ is not an inherent property of the sound waves constituting any particular utterance token, so too, the associated proposition comprising the *interpretation* of the utterance is not intrinsic to the abstract syntactic structure. Instead, the associated meaning is determined via arbitrary human convention, and the same syntactic item could just as well have had the interpretation currently expressed in English by ‘The rat is on the table’ or ‘The dog is on the hearth’.

In the context of classical computation, as above, one of the key constraints in the notion of an effective procedure is that the rules can be followed 'mindlessly', i.e. without knowing what the manipulated symbols are supposed to *mean*. As a consequence, there is no unique meaning determined by the procedure as such, and a multitude of distinct and *incompatible* interpretations are always possible. In the simple example given previously, a Turing machine ‘intended’ to compute the values of a particular truth function, say inclusive disjunction, can be easily reinterpreted as computing conjunction instead, simply by flipping our interpretation of the symbols ‘0’ and ‘1’. And the same procedure interpreted as computing conjunction could instead be construed as computing the values of the arithmetical function of multiplication. restricted to the numerical inputs 0 and 1. Yet *no causal nor counterfactual* features of the device have been altered by these reinterpretations, which indicates that neither of these factors is sufficient to ground claims concerning the purported ‘realist’ or non-observer-dependent status of computation in the physical world.

Similarly, formal systems in general are such that the transformations on symbols are not specified with reference to their intended interpretation. Many classical *negative* results in mathematical logic stem from this separability between formal syntax and meaning. The various upward and downward Löwenheim-Skolem theorems show that formal systems cannot capture intended meaning with respect to infinite cardinalities. As another eminent example, Gödel’s incompleteness results involve taking a formal system designed to be ‘about’ the natural numbers, and systematically reinterpreting it in terms of its own syntax and proof structure. As a consequence of this ‘unintended’ interpretation, Gödel is able to prove that arithmetical truth, an exemplary *semantical* notion, cannot, in principle, be captured by finitary proof-theoretic means.

In summary, there are *two* fundamental gaps separating formal procedures, standardly interpreted as computing the values of given functions, from the physical processes that we construe as implementing such procedures. First there is the gulf dividing the intended semantic interpretation from the bare syntactic formalism, and second there is the chasm between abstract syntactic formalism and physical reality. In *both* cases the gaps can only be bridged by an act of purely conventional human interpretation. And it is in this sense that computation in the physical world is inherently observer dependent.

7 COMPUTATION AND PRAGMATICS

I would now like to propose a different perspective on the issue. Rather than distinguishing ‘true’ from ‘false’ cases of implementation, what these and other proposed constraints do instead is to go some distance in distinguishing interesting, conceptually rich and *pragmatically useful* implementations from the many uninteresting, trivial and useless cases that abound in the space of possibility. It’s certainly true that there is no pragmatic value in most interpretive exercises compatible with ↑MR and ↑MR*. Ascribing computational activity to physical systems is *useful* to us only insofar as it supplies *informative outputs*, which in most cases will come down to new information acquired as a result of the implemented calculation.

So, interesting and useful observer dependent computation takes place when we can directly read-off something that *follows from* the implemented formalism, but which we didn’t already know in advance and explicitly incorporate into the mapping from the start. That’s the incredible value of our computational artefacts, and it’s the only *practical* motivation for playing the interpretation game in the first place. Hence a crucial difference between our computational artefacts and the attributions of formal structure to naturally occurring open systems, as employed by ↑MR* exercises, is that the mapping in the latter case is entirely *ex post facto* and thus supplies us with no epistemic gains. The abstract procedural ‘trajectory’ is already known and used as the basis for interpreting various state transitions in the open system and hence characterizing it as an implementation. In sharp contrast, we can use the intended interpretation of our artefacts both to *predict* their future behaviour, as well as *discover* previously unknown output values automatically.

And this is obviously why an engineered correlation obtains between fine-grained causal structure and abstract formal structure in the case of our artefacts – we want them to be informative and reliable! We also want them to be highly versatile, and this is where counterfactual considerations come to the fore in practice: over time we can do runs on a huge number of different inputs, and in principle the future outputs follow as direct consequences of the intended interpretation. So a physical system is *useful to us* as a computer only when its salient states are distinguishable by us with our measuring devices, and when we can put the system into a selected initial state to compute the output of our chosen algorithm on a very wide range of specific input values.

These *pragmatic* considerations supply clear and well motivated criteria for differentiating useful from useless cases of physical implementation. And I would advocate this type of pragmatic taxonomy in lieu of attempts to give overarching theoretical constraints purporting to distinguish ‘true’ from ‘false’ cases. Some basic desiderata for pragmatically valuable implementations include (a) fully automatic, (b) reliable, (c) versatile in the sense of computing values for a wide range of different inputs (d) non *ex post facto* (e) yielding increased predictive power with regard to future physical states of the implementing mechanisms, (f) possessing technologically manipulable initial configurations and output configurations detectable by our measuring devices and (g) physical rather than purely abstract constraints on the input and output characterizations.

8 CTM REVISITED

The last desideratum above supplies a relevant link to one of the opening themes of the paper, *viz*., defending CTM against SMA-based charges of empirical vacuity. When it comes to physical as opposed to purely abstract computation, we often want to place physical constraints on the characterization of inputs and outputs. In other words, the abstract inputs and outputs are given canonical physical interpretations. So the device must be such that it has certain types of causal powers to allow it to behave in the desired manner. For example, a stone or a bucket of water will never be able to pass the Turing test (TT), because the system lacks the appropriate causal powers. In order to pass, the computational device must produce *English sentences* as output, and Searle's wall can't do this. It may output some thermal energy that we could *further interpret* as code for the appropriate English sentence, but then we as observers are performing an extra step of interpretation which should not be required. And this 'test' could not be interactive like the TT unless the mapping from Searle's wall to the computational procedure were *not ex post facto* (if it were, then we would have to settle for canned 'exchanges' or sample dialogues computed after the fact). At least in the CRA, the set-up has the ability to interactively process the relevant physically specified input patterns and produce output in recognizable/readable Chinese syntax.

Hence I would advance a purely formal account of computation itself, as well as the SMA version of physical implementation, but still disagree with the view that MTC alone is sufficient to provide a full computational *theory of* particular subject disciplines, such as a *computational theory of vision*, or a *computational theory of the mind*. These are particular applications of MTC, and will require additional resources appropriate to the phenomena and subject areas in question. In this respect SMA is not in conflict with more relaxed (and empirically plausible) versions of CTM. What SMA directly threatens, and what has served as the traditional fulcrum in the dialectic, is the Computational Sufficiency Thesis (CST), which maintains that merely implementing a computational formalism of the appropriate sort constitutes a sufficient condition for mentality in the physical world.

It is salient to note that from a normal scientific perspective, CST is curious indeed. There are many different levels of description and explanation in the natural world, from quarks all the way to quasars. But there is no comparable sufficiency thesis in chemistry, biology, geology, astronomy. In other 'special sciences', membership in the corresponding level of description is a matter of degree and scientific utility, not a matter of some uniformly applicable sufficient condition, an essential or intrinsic property. 'Being a rock' is in no way an intrinsic property of the conglomerations of particles categorized as such, and this level of description is not captured by any simple sufficiency thesis. In turn, I would diagnose much of the controversy over CTM, the trivialization arguments, and concomitant defensive critiques of SMA to be engendered by ill advised allegiance to CST. The CTM camp places far too great a theoretical burden on computation alone. How could the mere fact of implementing the 'right' type of abstract procedure be enough to magically transform an insentient arrangement of matter and energy into a genuine cognitive system?

In contrast, I would argue that much more is required − the system must be anchored in and interact with the real world in a host of rich and multifaceted ways not satisfied by a mere stone or a bucket of water. In terms of a computationally based science of mind, a number of pragmatic and application-specific considerations should come to the fore, to critically augment the bare and global framework provided by MTC and SMA. Ideally, when treating the highly complex physical organism as implementing some abstract computational procedure, the ascribed formal structure should supply the high level *organizational key* for the underlying causal structure enabling the system to behave in the ways that it does, i.e. in the ways salient to its status as a *cognitive* system. Concomitantly, ascribed computational structure would then provide the high level (and empirically testable) key for *predicting* its future behaviour.

So, even if we were to grant (purely for the sake of argument and illustration) that the brain can be interpreted as implementing something like Fodor's [15] Language of Thought (LOT), still, this would *not* be an intrinsic property of the brain as a biochemical mechanism. Obviously, there would be no scientific interest in a mere *ad hoc* mapping from LOT onto the brain, although in principle this may be possible, *a la* ↓MR*. Instead, for a theoretically substantive approach, there would be a myriad of pre-existing and empirically intransigent 'wet-ware' constraints that the mapping would have to satisfy, in order to respect the salient causal structure of brain activity as discovered by neuroscience. The largely independent body of functional and anatomical data from neuroscience would supply a host of highly non-trivial restrictions on how the physical system itself is characterized and what the material state transitions should look like that are interpreted as implementations of the abstract computational procedures. A *scientifically significant* mapping is not free to view the arrangement of matter and energy comprising the human brain in terms of brain-irrelevant aspects such as cosmic ray bombardment, gravitational fields, arbitrary molecular kinetics, etc. Instead, it must restrict itself to salient *causal* factors pertaining to the physical system's time-evolution when viewed *as a brain*. So a version of Chalmers' causal regularities between states would in fact obtain in this more regimented and specialized case, because, like a standard computational artefact, the brain must perform the implemented computational procedures automatically and reliably.

If a physical system when viewed as a brain were methodically interpretable as implementing the LOT, this would entail that the transitions between the various neurological states instantiating respective tokens of mentalese symbols, obeyed a causal progression in accord with the transformation of these symbols as prescribed by the abstract computational formalism. *If* this could be done, it would provide a scientifically fruitful and explanatorily powerful key to organic cognition, because it would constitute a unifying perspective tying together actual brain function and the standard belief-desire framework of intentional explanation, as enshrined in the LOT.

This abstract computational interpretation of brain activity would also need to mesh with the input and output capabilities that we want to explain *via* the attribution of internal cognitive structure, e.g. intelligent linguistic performance as in a Turing test. So the computational level of description would have to conform with observable input and output patterns interpreted symbolically, as, say, sentences in an English conversation, to yield successful *predictions* of both new outputs given novel inputs, and predictions correctly describing new *brain configurations* entailed by the theory as realizations of the appropriate formal transformations required to produce the predicted symbolic output.

Such a project would have exceedingly non-trivial scientific/empirical value, not in the least undermined by Putnam-Searle type arguments. Objections of this kind have polemical force only against CST, and in light of the many *empirical* constraints and opportunities for testing predictions of both external behaviour and internal brain state, the CST would be rendered a completely gratuitous consideration. There is no single and simple sufficiency condition in this highly complex and multifaceted scientific enterprise. Merely implementing the LOT does not magically transform the brain into a mind. On this more scientifically plausible version of CTM, computation supplies the appropriate high level description of the brain for prediction and explanation of actual events, as well as an indispensible *bridge* between causally efficacious brain structure and high level accounts in terms of content bearing mental states. But computation alone does not make a mind. So I would argue that we should retain both MTC and SMA, *reject* CST, and embrace a more empirically grounded version of CTM. When faced with the triviality challenge that even a bucket of water could be interpreted as implementing the LOT, an advocate of this latter version of CTM could happily respond "Yes, and so what?".

REFERENCES

[1] Putnam, H., *Representation and Reality*, MIT Press, (1988).
[2] Turing, A., 'On Computable Numbers, with an Application to the Entscheidungsproblem', *Proceeding of the London Mathematical Society*, (series 2), 42, 230-265, (1936).
[3] Turing, A., 'Computing Machinery and Intelligence', *Mind*, 59: 433-460 (1950).
[4] Schweizer, P., 'Physical Instantiation and the Propositional Attitudes', *Cognitive Computation*, 4: 226-235 (2012).
[5] Searle, J., 'Minds, Brains and Programs', *Behavioral and Brain Sciences* 3: 417-424, (1980).
[6] Searle, J., 'Is the Brain a Digital Computer?', *Proceedings of the American Philosophical Association*, 64, 21-37, (1990).
[7] Bishop, J. M., 'Why Computers Can't Feel Pain', *Minds and Machines*, 19, 507-516, (2009).
[8] Fodor, J., 'The Mind-Body Problem', *Scientific American*, 24 (1981).
[9] Piccinini, C., 'Computation without Representation', *Philosophical Studies*, 137, 205-241 (2006).
[10] Chrisley, R. L., 'Why Everything Doesn't Realize Every Computation', *Minds and Machines*, 4, 403-420, (1994).
[11] Chalmers, D. J., 'Does a Rock Implement Every Finite-State Automaton?', *Synthese*, 108, 309-333, (1996).
[12] Copeland, J.,'What is Computation?', *Synthese*, 108:335-359 (1996)
[13] Block, N., 'Searle's Arguments against Cognitive Science'. In J. Preston and J. M. Bishop *Views into the Chinese Room*, Oxford University Press, (2002).
[14] Dennett, D. 'True Believers: the Intentional Strategy and Why it Works'. In A. F. Heath *Scientific Explanation: Papers Based on Herbert Spencer Lectures given in the University of Oxford*, Oxford University Press, (1981).
[15] Fodor, J., *The Language of Thought*, Harvard University Press, (1975).