(6694 words)

Dancing with Pixies:
Strong Artificial Intelligence and Panpsychism

Mark Bishop

In 1994 John Searle stated (Searle 1994, pp.11-12) that the Chinese Room Argument (CRA) is an attempt to prove the truth of the premise:

1: Syntax is not sufficient for semantics

which, together with the following:

2: Programs are formal,
3: Minds have content

led him to the conclusion that 'programs are not minds' and hence that computationalism, the idea that the essence of thinking lies in computational processes and that such processes thereby underlie and explain conscious thinking, is false.

The argument presented in this paper is not a direct attack or defence of the CRA, but relates to the premise at its heart, that syntax is not sufficient for semantics, via the closely associated propositions that semantics is not intrinsic to syntax and that syntax is not intrinsic to physics.[1] However, in contrast to the CRA's critique of the link between syntax and semantics, this

[1] See Searle (1990, 1992) for related discussion.

paper will explore the associated link between syntax and physics.

The main argument presented here is not significantly original – it is a simple reflection upon that originally given by Hilary Putnam (Putnam 1988) and criticised by David Chalmers and others.[2] In what follows, instead of seeking to justify Putnam's claim that, "every open system implements every *Finite State Automaton* (FSA)", and hence that psychological states of the brain cannot be functional states of a computer, I will seek to establish the weaker result that, over a finite time window every open system implements the trace of a particular FSA **Q**, as it executes program (**p**) on input (**x**). That this result leads to panpsychism is clear as, equating **Q** (**p**, **x**) to a specific Strong AI program that is claimed to instantiate phenomenal states as it executes, and following Putnam's procedure, identical computational (and *ex hypothesi* phenomenal) states (ubiquitous little 'pixies') can be found in every open physical system.

The route-map for this endeavour is as follows. In the first part of the paper I delineate the boundaries of the CRA to explicitly target all attempts at machine understanding – not just the script-based methods of Schank and Abelson (Schank & Abelson 1977). Secondly I introduce *Discrete State Machines, **DSM**s*, and show how, with input to them defined, their behaviour is described by a simple unbranching sequence of state transitions analogous to that of an inputless FSA. Then I review Putnam's 1988 argument that purports to show how every open physical system implements every inputless FSA. This argument is subsequently applied to a robotic system that is claimed to instantiate genuine phenomenal states as it operates.

Thus, unlike the CRA, which primarily concerns the ability of a suitably programmed computer to understand, this paper outlines a reductio-

---

[2] See Chalmers (1994, 1996a, 1996b) and also the special issue, *What is Computation?* of Minds and Machines, (vol.4, no.4, November 1994).

style argument against the notion that a suitably programmed computer qua performing computation can ever instantiate genuine phenomenal states. I conclude the paper with a discussion of three interesting objections to this thesis.

The Chinese Room

The twenty years since its inception have seen many reactions to the Chinese Room Argument from both the philosophical and cognitive science communities. Comment in this volume ranges from Bringsjord, who asserts the CRA to be "arguably the 20th century's greatest philosophical polarizer", to Rey who claims that in his definition of Strong AI, Searle, "burdens the [*Computational Representational Theory of Thought* (Strong AI)] project with extraneous claims which any serious defender of it should reject". Yet the CRA is not a critique of AI *per se* – indeed it is explicit in 'Minds, Brains, and Programs', as in other of his expositions, that Searle believes that there is no barrier in principle to the notion that a machine can think and understand. The CRA is primarily a critique of *computationalism*, according to which a machine could have genuine mental states (e.g. genuinely understand Chinese) purely in virtue of its carrying out a series of computations.

In the CRA Searle presented a rebuttal of the then computationalist orthodoxy that viewed cognition and intelligence as nothing more than symbol manipulation and search.[3] Following work on the automatic analysis of simple stories, a cultural context emerged within the AI community that appeared comfortable with the notion that computers were able to 'understand' such stories, a concept which can be traced back to the publication of Alan Turing's seminal paper 'Computing Machinery and Intelligence' (Turing 1950).

---

[3] E.g. Newell & Simon 1976.

For Turing, emerging from the fading backdrop of Logical Positivism and the Vienna Circle, conventional questions concerning 'machine thinking' were too imprecise to be answered scientifically and needed to be replaced by a question that could be unambiguously expressed in scientific language. In considering the metaphysical question, 'Can a machine think?', Turing arrived at the other, distinctly empirical, question of, whether, in remote interaction via teletype with both a computer and a human, a human could identify which was which as accurately as by chance. If so, the computer is said to have passed the *Turing Test*.

It is now more than fifty years since Turing published details of his test for machine intelligence and although the test has since been discredited by several commentators (e.g. Bringsjord 1992, Kelly 1993), the notion of a thinking machine continues to flourish. Indeed, the concept has become so ingrained in popular culture by science fiction books and movies that many consider it almost apostate to question it. And yet, given the poverty of current AI systems on relatively simple linguistic comprehension problems,[4] it is hardly surprising that, when writing on the subject, a phrase from Hans Christian Andersen slipped into Roger Penrose's mind (Penrose 1989). Nonetheless, throughout the 1970's and early 1980's, Searle and Hubert Dreyfus (Dreyfus 1972) remained isolated voices that surfaced above the hegemony of symbolically Strong AI. Still today, partly due to the (apparent) simplicity of its attack, the CRA is perhaps the best-known philosophical argument in this area.

In the CRA Searle argues that understanding of a Chinese story can

---

[4] As illustrated by the poor quality of the entrants to the annual Loebner prize (an award made to the program that can best maintain a believable dialogue with a human. See http://www.loebner.net/Prizef/loebner-prize.html).

never arise purely as a result of the state transforms caused by 'following the instructions' of any computer program. His original paper offers a first-person tale outlining how Searle could instantiate such a program, produce correct internal and external state transitions, pass a Turing Test for understanding Chinese, and yet *still* not understand a word of Chinese. However, in the twenty years since its publication, perhaps because of its ubiquity and the widespread background perception that, if it succeeds at all, its primarily target is *Good Old-Fashioned AI* (GOFAI), the focus of AI research has drifted into other areas: connectionism, evolutionary computing, embodied robotics, etc. Because such typically cybernetic[5] approaches to AI are perceived to be the antithesis of formal, rule-based, script techniques, many working in these fields believe the CRA is not directed at them. Unfortunately it is, for Searle's rule-book of instructions could be precisely those defining learning in a neural network, search in a genetic algorithm or even controlling the behaviour of a humanoid-style robot of the type beloved by Hollywood.

But what does it mean to genuinely understand Chinese? That it is not simply a matter of acting in the behaviourally correct way is illustrated if we consider Wittgenstein's illustration of the difference between following a rule and merely acting in accordance with it.[6] Although rule-following requires regularity in behaviour, regularity alone is not enough. The movements of planets are correctly described by Kepler's laws, but planets do not *follow* those laws in a way that constitutes rule-following behaviour.

It is clear that the CRA employs a similar rhetorical device. It asks: 'Does the appropriately programmed computer follow the rules of (i.e.

---

[5] Cybernetic AI is characterised by emphasis on 'sub-symbolic knowledge representation' and a 'bottom-up' approach to problem solving.

[6] Wittgenstein 1953, §§207-8, 232.

understand) Chinese when it generates 'correct' responses to questions asked about a story, or is it merely that its behaviour is correctly described by those rules?'. This difference between genuinely following a rule and merely acting in accordance with it seems to undermine Turing's unashamedly behaviouristic test for machine intelligence.

That the CRA addresses both phenomenal and intentional aspects of understanding and intelligence is clear from the introduction to Searle's original paper, where we find Searle's definition of Strong AI:

> But according to Strong AI the computer is not merely a tool in the study of the mind; rather the appropriately programmed computer really *is* a mind, in the sense that computers given the right programs can be literally said to *understand* and have other cognitive states. (Searle 1980, p.417 (p.67 in Boden)).

An axial statement here is that, 'the appropriately programmed computer really is a mind'. This, taken in conjunction with, '[the appropriately-programmed computer] can be literally said to understand' and hence have associated 'other cognitive states', implies that the CRA also, at the very least, targets some aspects of machine consciousness – the phenomenal infrastructure that goes along with 'really having a mind'.

However it is also clear from literature on the CRA that many philosophers do not believe that prestigious practitioners of AI take the idea of machine phenomenology and artificial consciousness seriously and hence that, in this aspect at least, the CRA is supposed to target a straw man. Yet several eminent cognitive scientists such as: Minsky, Moravec and Kurzweil have already speculated widely on the subject.[7] Further, as Searle makes

[7] See, e.g., Minsky (1985), Moravec (1988) and Kurzweil (1998).

clear, it was precisely such statements, emerging from a vociferous bunch of proselytising AI-niks discussing Schank & Abelson's work, that originally led him to formulate the CRA.

The idea that the appropriately-programmed computer really is a mind is eloquently outlined by Chalmers. Central to Chalmers' nonreductive functionalist theory of mind is the *Principle of Organisational Invariance,* ***POI***. This asserts that, 'given any system that has conscious experiences, then any system that has the same fine-grained functional organisation will have qualitatively identical experiences' (Chalmers 1996b, p.249). To illustrate the point Chalmers imagines a fine-grained simulation of the operation of the human brain – a massively complex and detailed neural network. If the outputs of each simulated neuron were identical to those found in a real brain, then, via 'Dancing Qualia' and 'Fading Qualia' arguments, Chalmers argues that the neural network must have the same qualitative conscious experiences as the brain.

What is clear from Chalmers, and indeed any of the prophets of computationally instantiated consciousness, is that the system's phenomenal states must somehow be realised by the formally-generated sequence of computational state transitions that arise as the program executes. But, following Turing, we must rid ourselves of a popular intuition:

> Importance is often attached to the fact that modern digital computers are electrical, and that the nervous system also is electrical. Since Babbage's machine was not electrical, and since all digital computers are in a sense equivalent, we see that the use of electricity cannot be of theoretical importance. (Turing 1950, p.439 (p.46 in Boden)).

Indeed, in 1976 Joseph Weizenbaum described a game-playing 'computer' that could be constructed from toilet rolls and coloured stones

(Weizenbaum 1976, pp.51ff.). Certainly functionalism, as a philosophy of mind, remains silent on the underlying hardware that causes computational state transitions – whether a program is executed on a PC or a MAC the results of its execution, the computational states it enters, are functionally the same.[8]

Discrete State Machines

In 'Computing Machinery and Intelligence', Turing defined DSMs as, "machines that move in sudden jumps or clicks from one quite definite state to another" (Turing ibid., p.439 (p.46 in Boden)), and explained that modern digital computers are a subset of them. An example DSM from Turing is that of a wheel machine that clicks round through 120^{o} once a second, but may be stopped by the application of a lever-brake mechanism. In addition, if the machine stops in one of the three possible positions it will cause a lamp to come on. Input to the machine is thus the position of the lever-brake, {*brake on; brake off*}, and the output of the machine is the lamp state, {*lamp on; lamp off*}.

[8] *Modulo* temporal constraints.

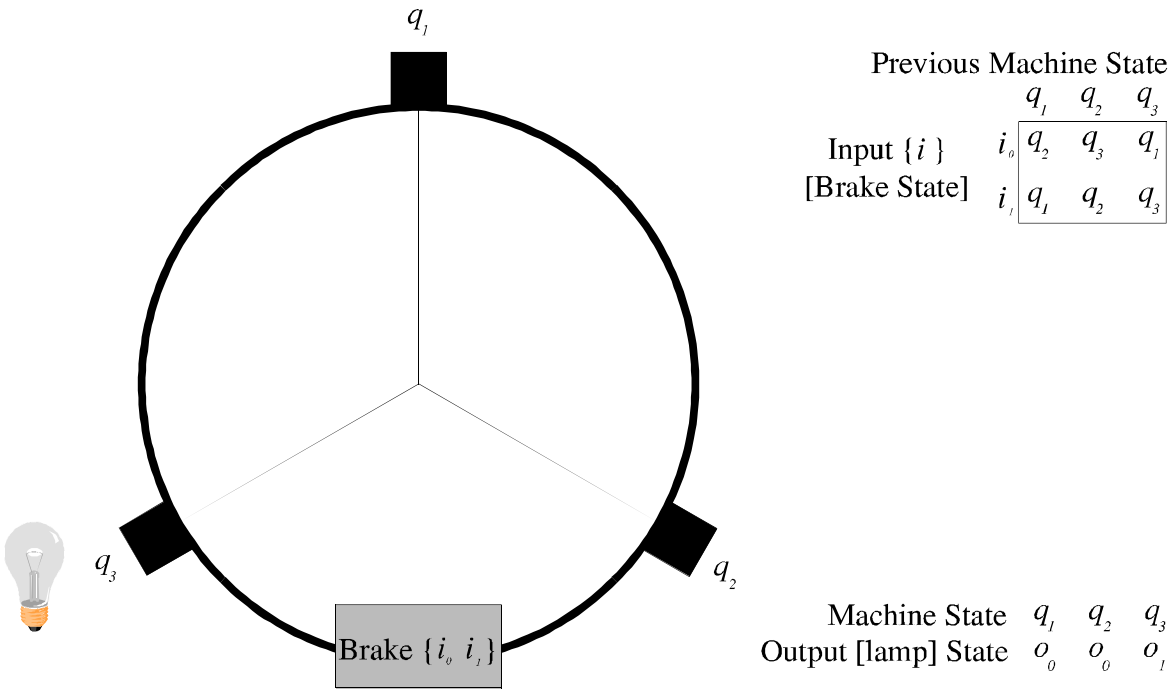

Figure 1:

Turing's 'Discrete State Wheel Machine'

Such a machine can be described abstractly in the following manner. Its internal [computational] state is labelled (arbitrarily) by a mapping function f that maps from the *physical state* of the machine (i.e. what position the wheel is in) to the *computational state* of the machine, $q \in \{q_1\ q_2\ q_3\}$. The input to the DSM, the brake position, is described by an input signal, $i \in \{\ i_0$ *brake off;* i_1 *brake on*$\}$. Hence the next state of the machine is determined solely by its current state and its current input as follows:

	q_1	q_2	q_3
i_0	q_2	q_3	q_1
i_1	q_1,	q_2	q_3

With its output being determined by:

State:	q_1	q_2	q_3
Output:	o_0	o_0	o_1 *[lamp on]*

Thus, with input to the machine either $\{i_0\}$ or $\{i_1\}$, (input undefined), the following branching state transition diagram describes the DSM's behaviour:

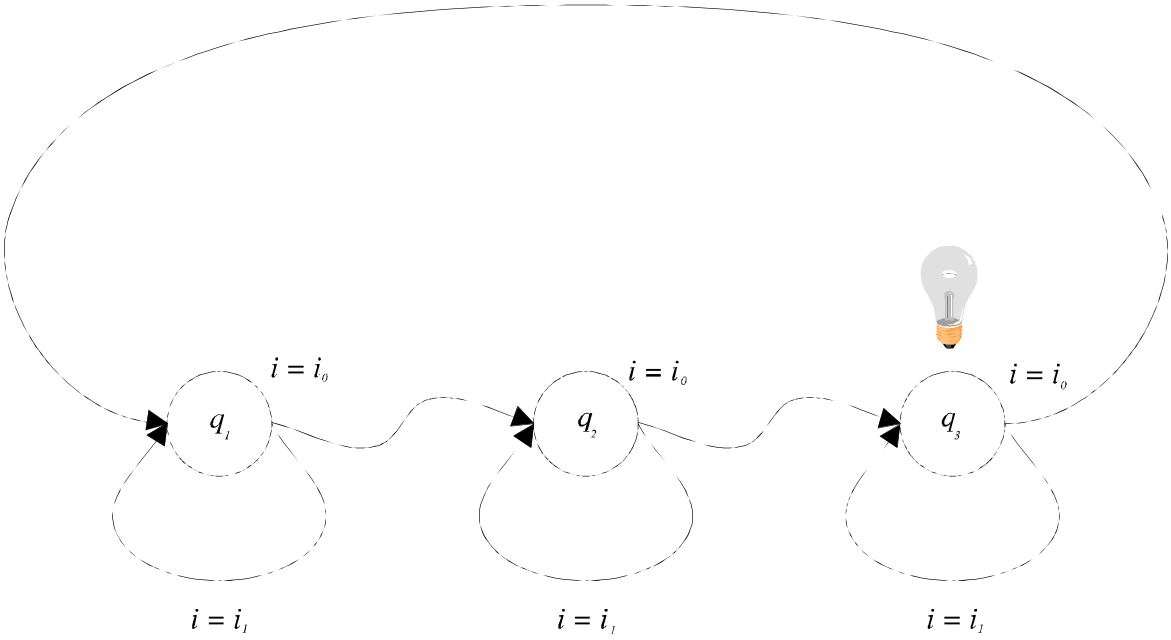

Figure 2:

State transition Diagram of Turing's Wheel Machine – input undefined

The above diagram has several branch points where the next state of the machine is determined by a state transition contingent on the current input. However, as shown below, for any specific input value there are no such branching state transitions. The machine's output (lamp on/off) is determined purely by its initial state, ($q \in \{q_1\ q_2\ q_3\}$), and the system input value, ($i \in \{i_0\ i_1\}$).

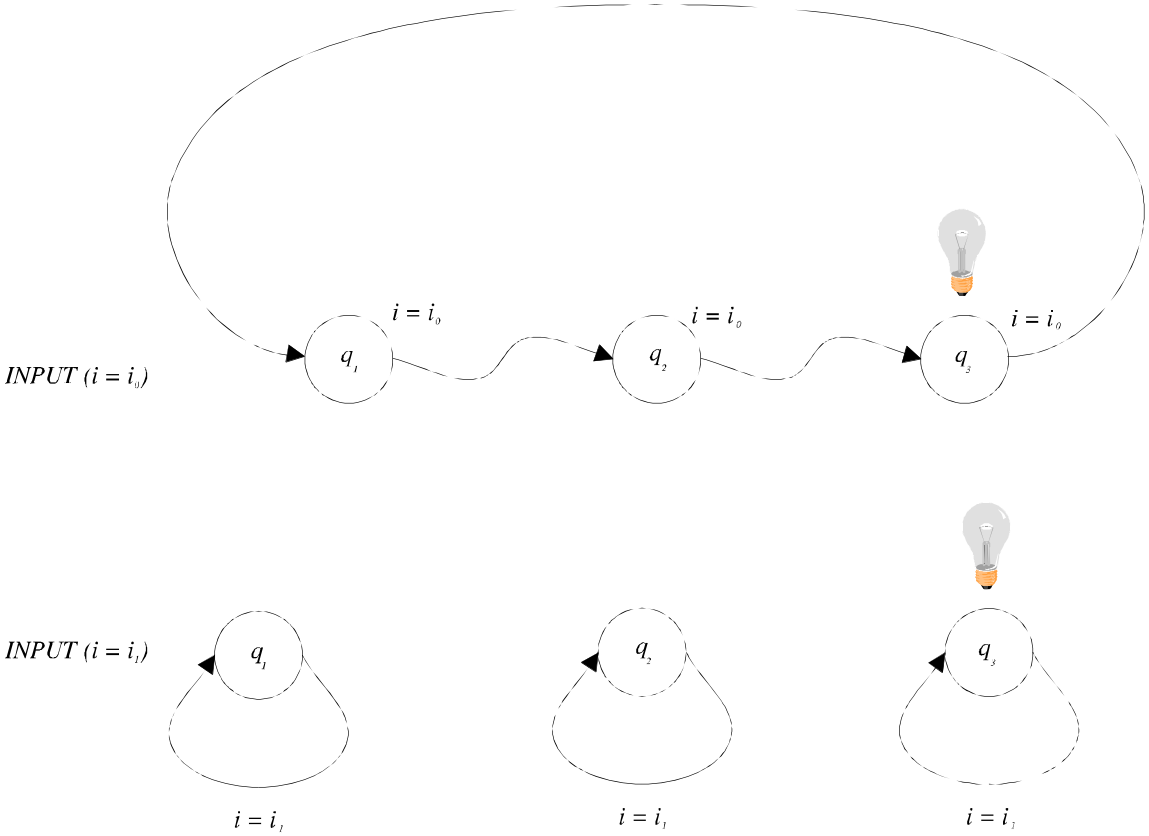

Figure 3:

State transition Diagram of Turing's Wheel Machine – input defined

Knowledge of the specific input to the machine's state transition table [program] has thus *collapsed its combinatorial structure*. Further, over a given time period, say $[t_1..t_7]$, all loops can be removed from the state diagram to form a linear path of state transits. The machine now functions, *like clockwork*, e.g.:

INPUT STATE 0:

$\langle q_1\ q_2\ q_3\ q_1\ q_2\ q_3\ q_1 \rangle$ OR $\langle q_2\ q_3\ q_1\ q_2\ q_3\ q_1\ q_1 \rangle$ OR $\langle q_3\ q_1\ q_2\ q_3\ q_1\ q_2\ q_3 \rangle$

INPUT STATE 1:

$\langle q_1\ q_1\ q_1\ q_1\ q_1\ q_1\ q_1 \rangle$ OR $\langle q_2\ q_2\ q_2\ q_2\ q_2\ q_2\ q_2 \rangle$ OR $\langle q_3\ q_3\ q_3\ q_3\ q_3\ q_3\ q_3 \rangle$

The following argument aims to show that for any Discrete State Machine

which, it is claimed, instantiates mental (phenomenal) states purely in virtue of its execution of a suitable computer program, we can generate a corresponding state transition sequence using any open physical system. That this conclusion leads to a form of panpsychism is clear, as if such state transition sequences are effectively found in most material objects, then phenomenal states must be equally ubiquitous.

Putnam's Claim

Hidden away as an appendix to Hilary Putnam's 1988 book Representation and Reality is a short argument that endeavours to prove that every open physical system is a realisation of every abstract Finite State Automaton and hence that functionalism fails to provide an adequate foundation for the study of the mind.

Central to Putnam's original argument is the observation that every open physical system, **S**, is in different *maximal states*[9] at every discrete instant and is characterised by a discrete series of *non-cyclic*[10] modal state transitions, $[s_1, s_2 .. s_t .. s_n]$. To simplify the following discussion of Putnam's claim and with minimal loss of generality[11], I will replace Putnam's arbitrary physical system, **S**, with a *counting machine*, generating the non-cyclic state sequence $[c_1, c_2 .. c_t .. c_n]$ in place of $[s_1, s_2 .. s_t .. s_n]$.

---

9 Putnam, (1988), p.122.

10 ibid., p.121.

11 Chalmers argues that for Putnam's open physical system to reliably transit a sequence of states it must include a natural clock (such as a source of radioactive decay), however Chalmers concedes that, "[p]robably most physical systems satisfy such a requirement" (1996a, p.316).

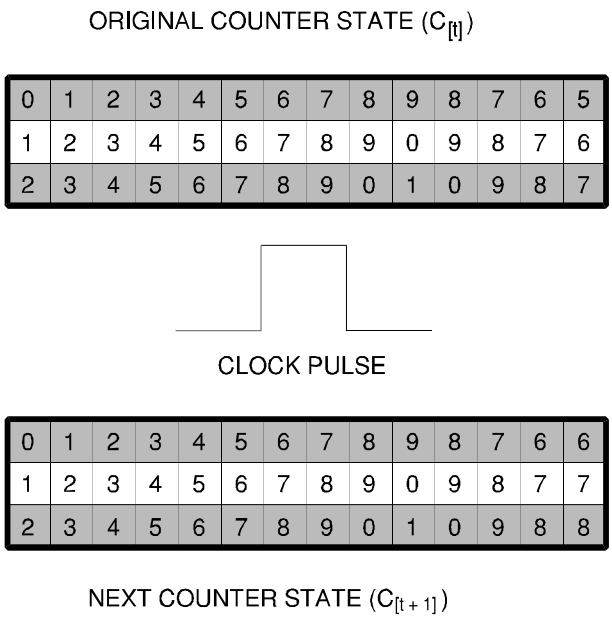

Figure 4:

A 'non-cyclic' Counting Machine

It is clear that given counter state [c_k] at time [t_k], it is trivial to predict its next state [c_{k+1}] at time [t_{k+1}]. NB. This transition from state [c_k] to [c_{k+1}] is both regular and carries full modal force – that the counting machine is in state [c_k] defines and contains the provision to force it to transit to [c_{k+1}] at the next clock interval.

Any inputless FSA is characterised by its state transition table, defining, given its current state, its subsequent state. Imagine, without loss of generality, that the state transition table for FSA **Q** calls for the automaton to go through the following sequence of states in the interval [t_1 .. t_6]:

<A B A B A B>

Next let us suppose we are given a counting machine, **C**, which goes through the sequence of states, [c_1, c_2, c_3, c_4, c_5, c_6] in the interval [t_1 .. t_6]. We wish to find a mapping between counter states [c_a] and [c_b] and FSA states [**A**] and [**B**] such that, during the time interval under observation, the counting

machine obeys **Q**'s state transition table by going through a sequence of states which the state mapping function will label [***A B A B A B***].

Putnam's Mapping

It is trivial to observe that if we map FSA state [**A**] to the disjunction of counting machine states, [$c_1 v\ c_3\ v\ c_5$], and FSA state [**B**] to the disjunction of counting machine states, [$c_2\ v\ c_4\ v\ c_6$], then the counting machine will fully implement **Q**. Further, given any counting machine state [c_a] $\in$ {c_1, c_3, c_5}, at time [t_a], we can predict it will enter state [**B**] at time [t_b].

To show that being in state [**A**] at time [t_1] caused the counting machine to enter state [**B**] at [t_2] we observe that at [t_1] the counting machine is in state [c_1], (which the mapping function labels FSA state [**A**]), and that being in state [c_1] at [t_1] causes the counting machine to enter state [c_2], (which the mapping function labels FSA state [**B**]) at [t_2]. Hence, given the current state of the counting machine at time [t], we can predict its future state and hence how the states of **Q** evolve over the time interval under observation.

Note, after Chalmers, that the counting system above will only implement a particular execution run of the FSA – there may be other state transition sequences that have not emerged in this execution trace. To circumvent this problem Chalmers posits a [counting] system with an extra dial – a sub-system with an arbitrary number of states, [$c_{[dial\text{-}state,\ counter\text{-}state]}$].

Now, as Chalmers suggests, we associate dial-state [1] with the first run of the FSA. The initial state of the counting machine will then be [$c_{[1,\ 1]}$] and we associate this with an initial state of the FSA. We then associate counting-machine states [$c_{[1,\ 2]}$], [$c_{[1,\ 3]}$] with associated FSA states using the Putnam mapping described earlier. If at the end of this process some FSA states have not come up, we choose a new FSA state, [**C**], increment the dial

of the counting machine to position [2] and associate this new state [$c_{[2,\ 1]}$] with [**C**] and proceed as before. By repeating this process all of the states of the FSA will eventually be exhausted. Then, for each state of the FSA there will be a non-empty set of associated counting machine states. To obtain the FSA implementation mapping we use Putnam's mapping once more and the disjunction of these states is mapped to the FSA state as before. Chalmers remarks:

> It is easy to see that this system satisfies all the strong conditionals in the strengthened definition of implementation [above]. For every state of the FSA, if the system is (or were to be) in a state that maps onto that formal state, the system will (or would) transit into a state that maps onto the appropriate succeeding formal state. So the result is demonstrated. (Chalmers 1996a, p.317).

Chalmers remains unfazed at this result because he states that inputless FSA's are simply an "inappropriate formalism" for a computationalist theory of mind:

> To see the triviality, note that the state-space of an inputless FSA will consist of a single unbranching sequence of states ending in a cycle, or at best in a finite number of such sequences. The latter possibility arises if there is no state from which every state is reachable. It is possible that the various sequences will join at some point, but this is as far as the 'structure' of the state-space goes. This is a completely uninteresting kind of structure, as indeed is witnessed by the fact that it is satisfied by a simple combination of a dial and a clock. (ibid., p.318).

But Putnam extends his result to the case of FSA's with input and output, by

arguing that an FSA with input and output is realised by every open physical system with the right input/output dependencies – if the physical system has the right input/output then it instantiates the FSA correctly. Patently this is a restriction on his original claim, but nonetheless it remains a significant result that, if correct, suggests that functionalism implies behaviourism.[12]

Putnam's original argument, using an open physical system to generate a series of non-repeating system states equivalent to the non-cyclic counting machine states described earlier, runs as follows. For any arbitrary FSA, take an open physical system with the right input/output dependencies, e.g. a rock with a number of marks on it encoding the input vector, (**x**), (where (**x**) encodes the finite set of input values, $\{x_1, x_2 .. x_n\}$) and another set of marks encoding the output vector (**o**)), (where (**o**) encodes the finite set of output values, $\{o_1, o_2 .. o_n\}$). Associate rock state $[s_1]$ at $[t_1]$ with the relevant initial state of the FSA, and rock state $[s_2]$ at $[t_2]$ with the subsequent state of the FSA etc. Putnam claims it is clear that by mapping each FSA state with the disjunction of associated rock states we ensure the system goes through the relevant state sequence $[s_1, s_2, s_3]$ that corresponds, using this mapping, to the relevant FSA state sequence, [***A***, ***B***, ***C***], with system output encoded by the other marks on the rock.[13]

However, as for Turing's DSM, the addition of input now makes the

---

12 Putnam (1988), pp.124-5. Any open system with the correct input/output dependencies implements the FSA with input/output. Hence every FSA with input (i) and output (o) is implemented by any physical system with the same input/output dependencies. Hence mentality is contingent only on input and output and functionalism implies behaviourism.

13 As before, we can use Chalmers' *extra dial* construction to ensure that all initially uninstantiated FSA states are generated by the system.

formalism non-trivial. There can now be branching in the execution trace, as the next FSA state is contingent upon its current state and the input. This gives the system a combinatorial structure. But, as Chalmers states, Putnam's revised construction does not fully encapsulate this structure – rather it merely manifests one trace of the FSA with a specific input/output dependency. So we are left with the counter intuitive notion that for example, when using say a rock to implement a *two plus two program*, we mark *two* on the input area of the rock and *four* on the output and credit the rock with computing the result . . .

In his 1996 paper, Chalmers introduces a more suitable FSA formalism, which makes explicit such input/internal-state dependencies, the *Combinatorial State Automaton,* ***CSA***. A CSA is like, (and no more powerful than), a conventional FSA except that its internal states, [**S**], are structured to form a set, $\{s_1, s_2 \ldots s_n\}$, where each element $\{s_i\}$ can take on one of a finite set of values or sub-states and has an associated state transition rule.

Chalmers then demonstrates how to map a CSA onto a physical system in such a way as to deal with such input/internal-state dependencies correctly and preserve the internal functional organisation of the original program, but only at the price of a combinatorial increase in the number of states required for the implementation. In fact, as he illustrates in his paper, executing even the most trivial FSA with input and output, over a small number of time steps would rapidly require a physical system with more states than atoms in the known universe to implement it. So it seems that, "we can rest reasonably content with the knowledge that the account as it stands provides satisfactory results within the class of physically possible system", and functionalism is preserved.

The problem that the CSA makes explicit is that of fully encapsulating the complex inter-dependencies between machine state and the input. To implement these using an open physical system requires an astronomical

number of internal states, whereas the simple implementation of an inputless FSA that Putnam describes functions only because of the subsequent loss of generality. However, as we observed with Turing's DSM, when input is defined over a specific time interval the combinatorial state structure collapses to a bounded linear path which can be simply generated using Putnam's mapping and any open physical system.

A Small Constriction

Consider a mobile robot whose behaviour is controlled by a program (**p**), acting upon input states (**x**), generating output states (**o**), running on computer hardware (**Q**). Consider the operation of the robot over a specified time interval $[t_1 \ldots t_k]$. Assume that after switch-on at time $[t_1]$, the robot experiences a series of phenomenal states until it is switched off at time $[t_k]$.

During the specified interval, $[t_1 \ldots t_k]$, the robot's input states are defined by data from its sensors forming the input set (**x**) = $\{i_1\ i_2\ i_3 \ldots i_k\}$ with its output states defining the actuator commands controlling its external behaviour, which together form the output set (**o**) = $\{o_1\ o_2\ o_3 \ldots o_k\}$. Let us now concede that the robot's control program, (**p**), instantiates a series of phenomenal states in **Q**, caused by interaction with its environment as the program executes. But what is it in the robot that manifests this property? Unless we allow mind to extend beyond the physical extent of the controlling computational hardware, **Q**, it must be manifested solely by it. ie. The claimed phenomenal properties of the system must be realised solely by **Q (p, x)**; CPU, **Q**, executing program (**p**), on input (**x**).

Now as **Q** executes (**p**) over the interval $[t_1 \ldots t_k]$, **Q** (**p**, **x**) generates a specific set of computational states, **S**, (**S** = $\{s_1, s_2 \ldots s_t\}$), at the discrete clock intervals of the CPU, **Q**. Due to the universal realisability of Turing Machine programs, the particular underlying computational engine, **Q**, is irrelevant to

the generation of the computational states $\{s_1, s_2 .. s_t\}$. Whether **Q** had at its heart a CPU made from toilet rolls or a 600Mz Intel Pentium P3, the associated computational states are the same yet it is the system's generation of these computational states that must result in its instantiation of phenomenal experience.

But we have already seen from Putnam how the computational states resulting from the execution of any given FSA, **Q**, (with specific input/output), can be mapped onto an open physical system, (e.g. a simple counting machine). Thus, by relaxing the requirement that the physical system instantiates the full combinatorial state structure of a program with general input, to the relatively trivial requirement that it just instantiate the correct state transitions for a specific execution trace, we sidestep the need for an exponential increase in state space. Over the specified time interval $[t_1 .. t_k]$ and with the input set defined, we are thus are able to replicate the computational states governing the robot's behaviour, (and hence the claimed phenomenal states), via an open physical system.

Three Objections: (1) Hofstadter: This is not Science

Douglas Hofstadter, in his 1981 critique of the CRA, objects that we can only perform a Putnam-style mapping *a posteriori*, i.e. we can only map the robot's computational states onto the physical states of the system after the program has executed and hence know the computational states it generates. Hence the Putnam construct is not a real mapping and this type of technique is 'not science'.

But Hofstadter is a little too harsh in his taxonomy of science and non-science. Unlike say, the intrinsic relationship between the solidity of a substance and the laws of physics, the relationship between the logical state of a computer variable and its physical implementation as a set of voltage levels is observer-relative. That is, in Searle's formulation, 'Syntax is not

intrinsic to physics' (Searle 1992, pp.207, 208). A computer variable cannot be uniquely identified from purely physical measurements without first knowing the mapping between the two domains (e.g. 5v = logic TRUE).

Following Turing's observations on universal realisability, if we repeat the robot's program execution with the same input on any suitable hardware, the resulting computational states are the same. Hence there is no substantive difference between equating physical system state [c_i] with FSA state [s_i], compared to equating the logical state TRUE with the physical state of +5v.

Clearly, once we know the computational states encountered in a given trace of a program, we can map them onto, and later read them off, the state transitions of an open physical system.

That Putnam's mapping can only be applied *a posteriori* is irrelevant to this discussion. Consider the experiment being repeated using the same FSA over the same length time interval [t'_1 .. t'_k], with the same input, [i'_1 .. i'_k]. The computationalist would continue to claim that the robot instantiated phenomenological states over this period. It is clear that *a posteriori* knowledge of system input does not impact upon this claim.

(2): The Execution of a Series of State-Transitions is not Sufficient to Attribute Phenomenal Properties to a Physical System[14]

This objection runs as follows. Putnam claims that his argument shows that any physical system realises any finite automaton's state transition table, but in fact it merely realises any desired sequence of states. Suppose an FSA recognises a simple (regular) language – given any input string, the machine enters a phenomenal state contingent upon the string being in the language or not being in it. It seems that because the automaton has gone through a

[14] Objection based upon a discussion with Peter Fletcher, lecturer in Computing & Mathematics at the University of Keele, UK.

sequence of states [s_1, s_2, . . . s_n], where each state causes the next, by using a Putnam mapping function, ***f***, to map from open physical system states to automaton states, the physical system can be viewed as going through the same sequence of states. Hence the physical system also 'recognises' the string. But this is a false intuition. Going through a certain sequence of states is not sufficient for a system to recognise a particular string, s. What matters is not just the sequence of states the automaton went through on this occasion, but the sequences of states it would have gone through if it had been presented with other strings.

This argument conflates two distinct properties: that of a system recognising the string s and that of a system experiencing phenomenal states. Input-sensitive counterfactual reasoning may or may not be a necessary property of any system of which it is claimed understands a language (and hence recognises the string s); however it does not constitute a necessary condition of any system that experiences phenomenal states – see below.

(3): Lack of Counterfactuals[15]

The open physical system described above does not genuinely replicate an FSA with input. In particular, it lacks the ability to correctly implement counterfactuals. As such it is not a full functional isomorph of an FSA system with input. Even though specific computational states may not be entered on a particular execution run of the FSA, the mere possibility that they could be if system input was different is required in any genuine functional isomorph of the FSA, and is a necessary condition for both systems to have identical phenomenal states.

The first point to note in this response is that it seems to require a non-

---

[15] Objection based upon a discussion with David Chalmers at the ASSC4 conference, Brussels, June/July 2000.

physical causal link between non-entered machine states and the resulting phenomenal experience of the system over the given time interval.

Secondly, we can use a type of Fading Qualia Argument (FQA)[16] to show that, in the context of FSA behaviour with input defined, counterfactuals cannot be necessary for phenomenal experience.

Consider the operation of two robots over the time interval $[t_1..t_k]$, with defined input $[i_1..i_k]$. One robot, R1, is controlled by a program designed to Chalmers' specification, replicating the fine-grained functional organisation of a system known to have phenomenal states; the second, R2, generates the computational states of R1 via an open physical system using Putnam's mapping. Hence, although the external behaviour of the two systems over the time interval is identical, for Chalmers, only R1 would experience genuine phenomenal states.

Yet even for R1, contingent on the defined input vector I, ($I = [i_1..i_k]$), only a small subset of potential machine states will be transited during the particular execution trace of state transitions, T_{R1} (I).

Now, with reference to system input, consider what happens if at each branch point in T_{R1} (I) we delete a state transition sequence that is not entered[17], then iteratively repeat this procedure until Chalmers' robot, R1, with full input-sensitivity, is step-by-step transformed into a second robot R2 whose behaviour is determined solely by a linear series of state transitions. We can imagine that throughout this replacement procedure R1 is repeatedly asked to report the colour of a red square placed within its sensor field.

Initially, R1 would both enter a phenomenal state corresponding to red,

---

16 Chalmers (1996b: p.255).

17 This is achieved by replacing one input-sensitive branching state transition, (cf. Figure 19.2), with a simple linear state transition contingent on the input, (cf. Figure 19.3).

and report that it perceived red. But what happens to the phenomenological experience of R1 as it incrementally undergoes the above transformation? In the spirit of Chalmers' exposition of the Fading Qualia Argument, imagine R1 to be at a basketball game, surrounded by shouting fans, with all sorts of brightly coloured clothes. Specifically, imagine R1 focusing on the bright red of the players' uniforms. Imagine also R2 in its final state (functioning as R1 without input sensitive branching behaviour, that is, simply performing a linear series of state transitions), being the same system but, by hypothesis, not experiencing any phenomenal states.

Between R1 and R2 there are a number of intermediate robots {R`} – what is it like to be them? As we transform R1 into R2, how does its phenomenal perception vary? Either its experience of phenomenal states must gradually fade (Fading Qualia) or it must switch abruptly at some point (Suddenly Disappearing Qualia). We can rule out the latter possibility by observing that it would imply that the removal of one such privileged branching state transition instruction would result in the complete loss of the robot's phenomenal experience.

Imagine then, that initially R1 was having bright red experience, which, as it transmutes to R2, must vanish. At some point R``s experience must stop being bright, yet the only difference between R1 and R` is that a sequence of non-entered machine states has been deleted.

It is clear that this type of fading qualia scenario is implausible, for otherwise we have a system, R`, whose phenomenal experience is contingent upon non-physical interactions with sections of its control program that are not executed – a form of dualism. Hence, if phenomenal states are purely physical phenomena, the phenomenal experience of the two robot systems, R1 and R2, must be the same.

Yet is this rendering of the FQA valid? David Chalmers has argued that

it is not[18]. In contrast to his version of the FQA, in this scenario although the first robot, R1, is sensitive to its input, the second, R2, is not ('it merely acts like a clockwork toy'). Yet it is clear that decreasing input sensitivity *per se* cannot affect R``s phenomenal experience, for consider what would happen to R1's qualia if the link between its frame store[19] and its visual sensor is damaged, such that its frame store constantly maintains a red image, irrespective of the colours processed by its optical sensor. This will result in R1, like R2, becoming insensitive to the colours of its visual input. When asked, R1 will now 'act like clockwork' and always report that the objects in its visual field are red, irrespective of their true hue.

This lack of input sensitivity will either deflate the phenomenal experience of R1 or have no effect (with any phenomenal states in the latter case analogous to the human experience of a red hallucination). But as R1's control program is unchanged and the data it reads from its frame store is of exactly the same form (a set of binary numbers), whether it is an accurate representation of the world or is erroneous, R1 will function as it always has and its phenomenal experience will be unchanged (i.e. constantly 'red').

Hence, in the execution of a computer program with known input, input-sensitive state transition branching behaviour (counterfactual reasoning) is not a necessary condition for phenomenal states to be instantiated by a computational system.

Conclusion

---

18 Discussion with David Chalmers at ASSC5 Durham, (North Carolina), May 2001.

19 A device than maintains a digital representation of an image obtained by a visual sensor such as a TV camera.

For any computing machine, **Q**, executing program (**p**), with known input (**x**), over the specified time interval [t_1 ..t_k], only a formal (and repeatable) series of state transitions occurs within its hardware. The generation of these state changes must be responsible for the generation of the machine's phenomenal properties. In this paper we have seen why, following Turing's observations on universal realisability, the underlying hardware that instantiates computational state transitions is unimportant and hence, following Putnam, that a series of such transitions could be implemented by any open physical system. Thus if, over a specified time interval, **Q** (**p**, **x**) has phenomenal awareness purely as a result of its execution of a Strong AI computer program, then so does any open physical system, and we find little pixies dancing everywhere . . .

References

Bringsjord, S. (1992) What Robots Can and Can't Be, (Dordrecht: Kluwer).

Chalmers, D.J. (1994) 'On Implementing a Computation', Minds and Machines, vol.4, pp.391-402.

(1996a) 'Does a Rock Implement Every Finite-State Automaton?', Synthese, vol.108, pp.309-333.

(1996b) The Conscious Mind: In Search of a Fundamental Theory, (Oxford: Oxford University Press).

Dreyfus, H. (1972) What Computers Cannot Do, (New York: Harper & Row).

Hofstadter, D. (1981) 'Reflections', in The Mind's I: Fantasies and Reflections on Self and Soul, (eds.) D.Hofstadter & D.C.Dennett (London: Penguin), pp.373-382.

Kelly, J. (1993) Artificial Intelligence: A Modern Myth, (Chichester: Ellis Horwood).

Kurzweil, R. (1998) The Age of Spiritual Machines: When Computers Exceed Human Intelligence, (New York: Viking).

Minsky, M. (1985) The Society of Mind, (New York: Simon & Schuster).

Moravec, H.P. (1988) Mind Children: The Future of Robot and Human Intelligence, (Cambridge, MA: Harvard University Press).

Newell, A. & Simon, H.A. (1976) 'Computer Science as Empirical Enquiry: Symbols and Search', Communications of the ACM, vol.19, pp.113-26.

Penrose, R. (1989) The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics, (Oxford: Oxford University Press).

Putnam, H. (1988) Representation and Reality, (Cambridge, MA: MIT Press/Bradford Books).

Schank, R.C. & Abelson, R.P. (1977) Scripts, Plans, Goals & Understanding, (Hillsdale, NJ: Lawrence Erlbaum).

Searle, J.R. (1980) 'Minds, Brains, and Programs', Behavioural and Brain Sciences, vol.3, pp.417-424.

(1984) Minds, Brains and Science, (London: BBC Publications).

(1990) 'Is the Brain a Digital Computer?', Proceedings of the American Philosophical Association, vol.64, pp.21-37.

(1992) The Rediscovery of Mind, (Cambridge, MA: MIT Press).

(1994) The Mystery of Consciousness, (London: Granta Books).

Turing, A.M. (1950) 'Computing Machinery and Intelligence', Mind, vol.49, pp.433-460.

Weizenbaum, J. (1976) Computer Power and Human Reason: From Judgement to Calculation, (San Francisco: W.H.Freeman).

Wittgenstein, L. (1953) Philosophical Investigations, (Oxford: Blackwell).