From Chinese Room to Human Window

Maarten van Emden [*] *André Vellino* [†]

ABSTRACT

The debate in philosophy and cognitive science about the Chinese Room Argument has focused on whether it shows that machines can have minds. We present a quantitative argument which shows that Searle's thought experiment is not relevant to Turing's Test for intelligence. Instead, we consider a narrower form of Turing's Test, one that is restricted to the playing of a chess endgame, in which the equivalent of Searle's argument does apply. An analysis of time/space trade-offs in the playing of chess endgames shows that Michie's concept of Human Window offers a hint of what a machine's mental representations might need to be like to be considered equivalent to human cognition.

keywords: Turing Test, Chinese Room, Human Window, computer chess, time-space trade-offs, Pareto optimality

1. INTRODUCTION

The literature in cognitive science, philosophy, and Artificial Intelligence has seen considerable debate about the computational theory of mind as a result of Searle's Chinese Room Argument (for a sampling, see [1]). Searle's argument was intended to demonstrate that machines cannot have minds in the same sense that people do. Although Searle [19] does not directly address the Turing Test, his argument can reasonably be construed to apply to it, as argued by [8].

In this paper we address some of the confusions that have fueled this debate, and concentrate on the fact that the Chinese Room Argument can be applied to a restricted cognitive problem — the playing of a chess endgame. Here we consider a variety of implementation strategies and criteria for evaluating them. Among these strategies is a minimal cost criterion and Michie's criterion [13] of whether they fit in the "Human Window".

Turing may have anticipated that human beings are too easily fooled, as ELIZA [24] subsequently showed, by a program that simulates a chat between strangers without any common interest. For Turing the adequate performance of linguistic behaviour was not the only distinguishing feature of intelligence: solving a chess problem and, especially, learning, were also demarcation criteria. Accordingly, Turing [22] suggested as a suitable question for the test

> *I have K at my K1, and no other pieces. You have only K at K6 and R at R1. It is your move. What do you play?*

However, the ability to solve a chess problem, as demonstrated by the success of contemporary chess playing programs [4], may not be sufficient to demonstrate the presence of cognition. We show in the following section that the restriction of this task to the endgame may be performed in the manner of Searle's Chinese Room; that is, without cognitive ability. For chess endgames, we argue, the Chinese Room argument is valid. In [13] Michie rejects the Chinese Room type implementations of chess endgames on different grounds that suggest alternative models for understanding human cognition.

[*](Corresponding Author) Department of Computer Science, University of Victoria, Victoria, Canada. vanemden@cs.uvic.ca
[†]André Vellino, School of Information Studies, University of Ottawa, Ottawa, Canada. avellino@uottawa.ca

2. TURING'S TEST AND THE CHINESE ROOM

To make less vague the meaning of such words as "intelligence" and "thought", A.M. Turing [22] proposed in 1950 a test to determine whether a suitably programmed computer can be said to think. The test consists of a game, the Imitation Game, between two players, a human who we call the "interrogator" and another player who is either a computer program or a human. The players play by exchanging messages. The goal of the interrogator is to determine whether his correspondent is human. The difficulty here is that, on the one hand, the parties need to be strangers to each other, yet be motivated to chat[‡]. If this is actually the case, this person cooperates, welcoming the opportunity to converse with a fellow human. We call the interrogator's messages "challenges"; what he receives we call "responses". The only way the interrogator can determine whether the other party is human is by the content of the text messages interchanged. If the interrogator can do this sufficiently reliably, then he has won the Imitation Game. Otherwise, the computer program has passed the "Turing Test" and Turing regarded it capable of thought.

Presumably, a prerequisite for the computer's success in the Imitation Game is an understanding of English, or whatever language is chosen for the conversation. The most widely discussed objection to the possibility of computer understanding of a human language was given by Searle [19]. Searle's argument has been named the Chinese Room, after its most memorable features: the Chinese language and a closed room. Searle aimed to refute "the claim that [an] appropriately programmed computer literally has cognitive states" [19] by addressing the work of Schank [18] who used scripts as vehicles for the computer understanding of simple stories. However, Searle claims that his argument also applies to Winograd's SHRDLU [25], Weizenbaum's ELIZA [24], "or indeed any Turing machine [sic] simulation of human mental phenomena". Accordingly, we can divest Searle's version of the Chinese Room of its complications necessitated by Schank's work and simplify it so that it fits Turing's Imitation Game. Searle's original formulation is quoted in Appendix A.

In this simplified version of the Chinese Room Argument, a Chinese interrogator exchanges messages with a room occupied by a person who knows no Chinese. The person is handed each challenge, in Chinese, by the interrogator as text on a slip of paper. The person has access to a library that contains, for every challenge, a suitable response in Chinese. The library's contents are such that the interrogator receives responses such as a human might give. The interrogator will lose the Imitation Game by concluding that he is conversing (in Chinese) with a human. As the person executes only a simply automatable information-retrieval function, the Chinese Room is trivially implementable as a computer program, as Searle would presumably claim. According to Turing's criterion such a program is capable of thought, whereas clearly it is not. Thus Searle claims to have refuted the passing of Turing's Test as a criterion for thought.

3. THE CHINESE ROOM IS IRRELEVANT

Although we have our own reservations about the significance of the test proposed by Turing, we will restrict ourselves in the present paper to a criticism of Searle's argument.

Searle overlooked the fact that Turing assumed that a computer program that passes the test runs on an *actual computer*. Admittedly not a 1950 computer, but one that could be built within, say, fifty years from the date of publication. But we can be sure that Turing did not envisage a computer that can only exist in the imagination of a professor of philosophy. We will show that a computer able to pass the Turing Test would have to have more interesting capabilities than the Chinese Room. Whether these capabilities are relevant to cognition is a question we shall return to the philosophers.

We would never have heard of the Chinese Room if Searle had made a back-of-the-envelope calculation along the following lines. The challenges by the interrogator are not restricted to a few words. Moreover, the interrogator does not emit isolated challenges, but tries to strike up a conversation, requiring the computer program to respond to a challenge in the context of past answers. Thus the response is not only determined by the challenge, but also by some sort of state of the conversation. Suppose, favourably for the Chinese Room argument, a safe lower

[‡]Michie [15] notes the difficulty in the following passage: "Had Turing covertly substituted himself for the machine in such a test, examiners would undoubtedly have picked him out as being a machine. A distinguishing personal oddity of Turing's was his exclusive absorption with the literal content of spoken discourse. His disinclination, or inability, to respond to anything in the least 'chatty' would leave the honest examiner [interrogator] with little alternative but to conclude: 'this one cannot possibly be the human; hence the other candidate must be'."

bound of 50 for the average number of words of the challenge plus the state description. Suppose, favourably for the Chinese Room argument, that the correspondents have only the rather small vocabulary of 1024 words. The number of sequences of length 50 from such a vocabulary is $1024^{50} = 2^{500}$.

This does not of course mean that the information-retrieval system has to contend with that number of items. There is a great deal of redundancy resulting from constraints from several sources: grammar, semantics, and context. Such redundancy can be reliably estimated. According to Shannon's findings ([20], page 26), this redundancy is less than $3/4$, to use a safe upper bound. A redundancy of $3/4$, which we now assume, is considerable from the usual point of view: it implies that such texts can be compressed by a factor of four. Therefore it reduces the size of the database by a factor of four. That is, the size of the database is reduced from 2^{500} to 2^{498}.

That latter number is still many orders of magnitude larger than any *physically* relevant number — for instance the number of elementary particles in the universe. Since candidates for passing Turing's Test must be programs running on *physically* realizable computers, it is clear that Searle's Chinese Room fantasy is not relevant to Turing's claim for his test, whatever the merits of that claim may be.

On the basis of the above back-of-the-envelope calculation one may be tempted to completely dismiss Searle's idea. In the remainder of this paper we show that Searle rediscovered an idea that is fruitful in a related context, where it offers a hint about what form cognition may take in humans or in computers.

4. DECOUPLING THE IMITATION GAME FROM THE TURING TEST

In view of the extraneous factors preventing success at the Imitation Game from being a reliable indicator for thought in the human sense [9], it is wise to consider alternatives. Chess endgames have several advantages. They pose a challenge to human players and, before the advent of computers, they have inspired chess researchers to produce a scholarly literature. Hence it cannot be denied that success at chess endgames exercises human thought at several levels in several modes[§].

Chess endgames suggest a stripped-down version of the Imitation Game. The interrogator's challenges are restricted to positions in a chess endgame. The human correspondent is a chess master and is restricted to responses in the form of an optimal move, as is the computer program.

A difference between chess endgames and the natural language Imitation Game is that, for chess endgames, Searle's idea of the Chinese Room is feasible. In fact, in his 1970 PhD thesis [21], T. Ströhlein anticipated this idea by a complete position/move database for at least one chess endgame. In a more accessible publication in 1977, Clarke [5] reported having constructed a database of 98,304 entries to allow a computer program to play a perfect game of King and Pawn versus King (KPK). To be precise, these authors paired each position with an optimal move from that position. If we apply Searle's Chinese Room Argument to this database look-up strategy for solving KPK endgames, Searle would no doubt claim that no thought is involved when this computer program plays its perfect endgame. With this we agree. And, when humans play chess, moves are typically generated by a cognitive process.

A chess endgame differs from the natural-language Imitation Game in that it can also be implemented in a small program that encodes the rules of chess and traverses the corresponding game tree. This is the opposite of the database approach in that it requires a small amount of memory and a large amount of processing power[¶].

We have now identified two ways of implementing a chess end game such as King and Pawn against King: distinguished by whether it uses the minimum amount of processing versus requiring a minimum amount of memory. The first alternative consists of a database with an entry for each individual position. The processing is then limited to a single look-up. We call this alternative the *extensional extreme*. The second alternative we call the *intensional* [‖] *extreme*. Are there alternatives to these extremes?

[§]In Michie's words [12]: "A reason for being interested in games is that they provide a microcosm of intellectual activity in general. Those thought processes which we regard as being specifically human accomplishments — learning from experience, inductive reasoning, argument by analogy, the formation and testing of new hypotheses, and so on — are brought into play even by simple games of mental skill. The problem of artificial intelligence consists in the reduction of these processes to the elementary operations of arithmetic and logic."

[¶]When we say "program" for the computer participant in the Imitation Game, we include the database (if used) as well as the executable logic.

[‖]"Extensional" by analogy with predicate logic, where the extension of a predicate is the set of tuples of individuals for which the predicate is true. "Intensional" for no better reason than that, independently of logic, it sounds like the opposite of "extensional".

We mentioned Clarke's choice [5] of end game because of his early publication. Around the same time the more difficult King and Queen against King and Rook (KQKR) was implemented in the same way by Kenneth Thompson of Bell Labs. As its database has about 4 million entries, and as it was tested in an interesting way (see later quote from Michie [13]), we choose it as representative for this class of programming problems.

One approach is to start at the database end — the extensional extreme — and to make it more compact. The database contains pairs consisting of a board position and the corresponding move for optimal play. One can extract from the database the set

$$pos(m) = \{p \mid \langle p, m \rangle \text{ is in the database}\}$$

for any position m. Using techniques of machine learning one can find position features that characterize $pos(m)$ [16]. Then, for given m, one can replace $pos(m)$ by a rule that tests a position on these features and advises move m if these features are present. If we do this for all moves m, then we get a rule base (which we regard as part of the program) that is much smaller than the original database. The result is a much smaller program that requires more processing than the simple database look-up, including at least the additional processing for classifying the position. The additional processing is partly offset by the fact that the rule base is now much smaller.

Starting from the intensional extreme, we can also obtain improvement. This extreme requires a large amount of processing because it needs to traverse a significant part of the game tree, symbolized by the tree on the left in Figure 1.

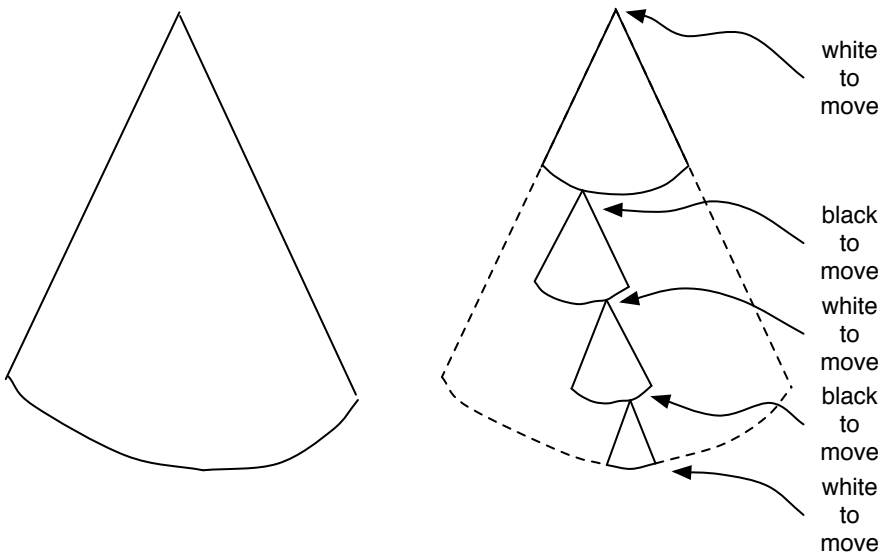

Figure 1: Symbolically, for comparison, the entire game tree for KQKR (left) and the game trees for successive subgames (right).

A chess expert can suggest a decomposition of KQKR into a sequence of subgames. In each subgame, winning can be defined as having reached a certain goal, which is a set of positions each of which has the property that it is easier to win the remainder of the KQKR game from such a position. This remainder may need to be decomposed in the same way. This is the approach taken by Michie with his Advice Language [11].

For each of these subgames the game tree is shallower than for KQKR. As the size of the game tree increases exponentially with depth, the total size of the game trees for the subgames is much smaller than the one for the entire game of KQKR; see the tree on the right in Figure 1.

Which subgame to play depends on the position. We consider for each subgame sg the set of positions

$$pos(sg) = \{p \mid sg \text{ is the subgame appropriate in position } p\}$$

This approach is valuable because it is possible to characterize, for each sg, the set $pos(sg)$ using a few easily computable criteria. Thus KQKR can be implemented by rules that pair the position criteria with the appropriate

subgame. This approach leads to a program that requires more memory than the most compact one and much less computing power. So much less in fact, that it runs quite quickly on an inexpensive laptop computer.

We have identified four kinds of implementation of KQKR, as shown in Figure 2.

1. The extensional extreme: a program for retrieving from a database consisting of individual $\langle position, move \rangle$ pairs. This has maximal memory requirement.
2. The intensional extreme: a program for traversing the entire KQKR game tree. This has maximal requirement in terms of processing power.
3. A program based on rules pairing position classes with moves.
4. A program based on rules pairing position classes with subgames and then, for each subgame selected, traversing the corresponding game tree.

Figure 2: Four types of KQKR implementation.

Items 3 and 4 can be implemented in a variety of ways.

5. PARETO-OPTIMALITY

Let us consider the universe of all computer programs that perform a given task. We characterise it by two coordinates: the number of processor cycles required for completion and the number of bytes required for data and working storage. The two coordinates allow us to plot each program as a point in a plane, as we do in Figure 3.

The efficiency with which these programs perform the task is judged according to these twin criteria. If an implementation P is represented by a point that is below and to the left of that representing implementation Q, then the latter is of no interest: P is faster without requiring more memory or requires less memory without being slower, or both. Because there is not a single criterion, it is not possible to identify a single program that is most efficient.

As the task to be performed is fixed, the points representing the programs cannot come arbitrarily close to the origin, which represents the empty set of programs for the given task requiring neither processing nor memory. The set of points in the diagram that are not dominated by any other points represent the programs that are, each in their own way, most efficient. These points are circled in Figure 3. A situation in which optimization is required according to multiple criteria is common in economics. Here the points that are not dominated are called *Pareto-optimal*. The totality of such points we call the *Pareto-frontier*.

An important property of the frontier is that it is curved toward the origin. This is not a property of KQKR specifically, but more generally of tasks for which the intensional and extensional extremes are far apart. For such a task, the compact implementation can be speeded up in a generic way, for instance by memo-ization**. The result is a faster and less compact algorithm. The program can be made still faster by the same process where the amount of memory is made available for the memo-ization database is enlarged by the same proportion. The relevant phenomenon here is *that larger and larger such databases are subject to a law of diminishing returns*. This results, on a logarithmic scale, in a curve toward the origin at the upper left part of the frontier.

The extensional extreme can also be improved in a generic way, namely by data compression. This process makes the database more compact at the expense of processor cycles needed to decode the compressed data. One can apply more and more sophisticated data compression methods. These are, again, subject to diminishing returns: to achieve the proportionally equivalent compression one needs more additional computation. On a logarithmic scale, the diminishing of the returns translates to a curving of the frontier in the direction of the origin at the lower right-hand end of the frontier.

So far we have only been able to eliminate the extreme implementations from considerations: there is no objective way to prefer the hybrid options (3) or (4) in Figure 2. This is because we have only the considered the *demand*

**Due to Michie [10]. By now his invention has become so widely used that it has passed into anonymous computer lore. See, e.g. [6].

side of KQKR implementation. On the *supply* side we need to consider computer systems capable of running the implementations considered. Such systems vary enormously in processor power and memory size. Thus we can compare computer systems that cost less than a given amount by placing them in a continuum characterised by two coordinates: processor speed and number of bytes of memory.

In Figure 3 the points cannot get arbitrarily close to the origin: the given task cannot be accomplished with zero processor cycles and zero memory. Here the situation is the dual of the one in Figure 4: the points cannot get arbitrarily far away from the origin: for a given cost there is a limit to the amount of speed and to the amount of memory that can be obtained. Again we can disregard computer systems that are dominated by those that do not exceed the cost limit and are faster or have more memory. Again there is a Pareto-frontier of non-dominated points, and these represent the computer systems that are Pareto-optimal. We assume similar technology across computer systems, so that there is a more or less constant memory versus processor power trade-off. On a logarithmic scale this implies that the frontier is curved away from the origin.

Consider the Pareto-frontier of computer systems of given cost and superimpose it on the Pareto-frontier for KQKR[††]. For a sufficiently low cost the frontiers for that cost and for KQKR do not intersect. This means that KQKR cannot be implemented for that low cost. However, in an idealized market there is a cost at which the frontiers just touch. As the frontier of KQKR is curved towards the origin, and as the frontier for the computer systems of the given cost is curved away from the origin, the point at which the frontiers touch is nearer to implementations (3) and (4) than to extensional and intensional extremes.

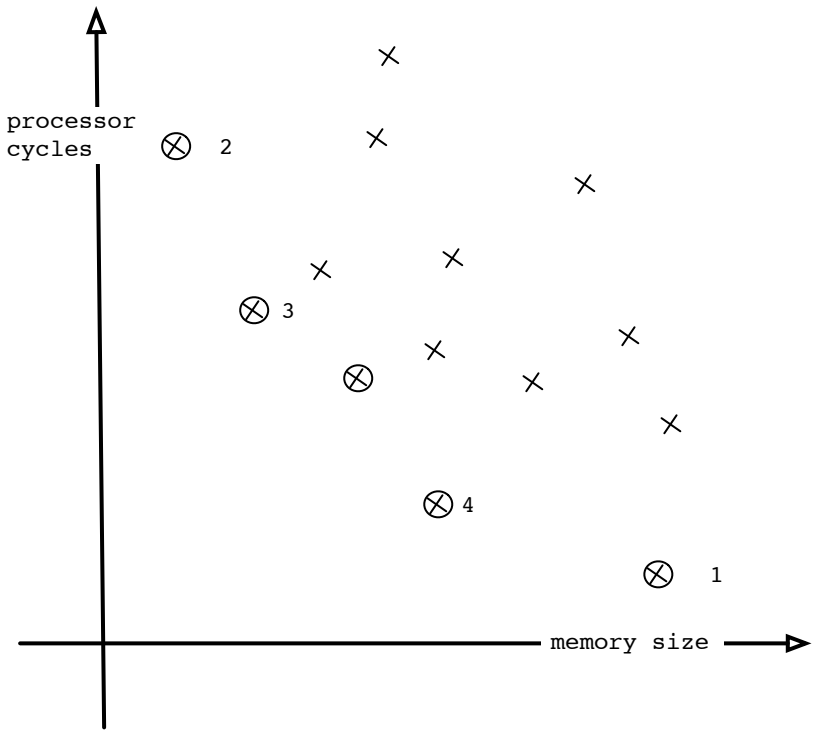

Figure 3: Trade-off frontier for KQKR.

We have encountered the curious situation that introducing cognitively significant concepts in the KQKR task saves dollars. [‡‡]

[††]For the sake of argument we suppose an idealized market where arbitrarily cheap computers are on offer, with correspondingly low speed or small memory. In the actual market even the cheapest computers have enough speed and memory to accommodate Pareto-optimal implementations of all the chess endgames studied in the literature.

[‡‡]From the point of view of Darwinian Natural Selection, however, this result is perhaps not surprising: one would expect the evolution of a conscious problem solving capability to result in a relatively optimal use of resources.

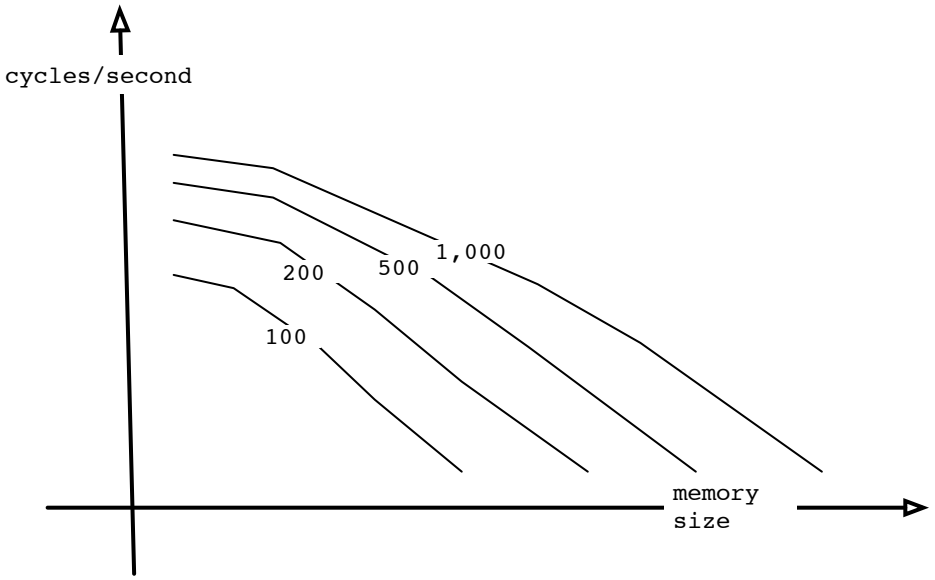

Figure 4: Processor power versus memory trade-off for computer systems. The lines connect systems of equal cost. The numbers count unspecified units of cost.

6. THE HUMAN WINDOW

We have considered cost as criterion for the implementation of chess endgames, trading off processor power against database size. Michie [14] proposed another criterion for evaluating software: whether their operation is *intelligible by humans*. He pointed out that the extensional extreme is meaningless to a human: each database entry asserts a move for a position without any indication as to why or how this move would contribute to the goal of mating the opponent. His concerns are best introduced by the following quote [13]:

> *. . . a computer chess experiment at the recent [1977] IFIP meeting at Toronto. Kenneth Thompson of Bell Laboratories had computed out the entire problem space for King and Queen against King and Rook, and had represented the optimal strategy for both sides in the form of a look-up table in machine memory. The table comprises about 4 million entries corresponding the the total number of legal King-Queen-King-Rook positions after excluding symmetrical cases. For all but a few special starting positions this ending is a theoretical win for the Queen's side. It is also known that chess-masters can ordinarily guarantee to execute the win against any human opponent. At Toronto two International Masters, former World Correspondence Chess champion Hans Berliner and Canadian champion Lawrence Day, were invited to demonstrate the winning strategy in play against the optimal defense embedded in Thompson's database. To their embarrassment they discovered that they could not win against the machine in spite of many renewed attempts.*
> *Naturally they then wanted to ask the program what were the key ideas underlying its defensive strategy. Of course, neither the program nor its author could answer such a question, since the program's knowledge, although comprehensive, had not been condensed into a form which humans can mentally handle. Since it does not contain things like ideas, concepts, themes, goals and the like, it is in no position to give answers like: "At this stage White must drive the enemy King on to the edge of the board". The program has no conceptual interface.*

Could (3) and (4) in Figure 2 be said to constitute a conceptual interface?

The intensional extreme consists of nothing more than the rules of chess and the statement that what is good for one player is bad for the other. Such a program is easy for a human to check for correctness. However, using it requires following through many layers of moves and countermoves to the end of the game. Beyond a few such

layers human short-term memory becomes inadequate. In endgames most positions are many layers away from the end. Michie characterizes the intensional extreme as *human-intelligible*, but *not human-executable*.

The extensional extreme is, on the other hand, human-executable. It could be printed in the format of a telephone directory. It would then be a simple operation to look up any position and find the corresponding optimal move. But the bare statement of the optimal move is not human-intelligible. As the above quote shows, it may actually seem wrong to a chess master.

Alternatives (3) and (4) in Figure 2 suggest that, in Michie's words ([14], page 109),

> *there exists a spectrum of equally complete and correct machine representations, differing according to the balance struck between calculation and memory. The optimal compromise is determined by the relative costs of these two commodities.*

In the spectrum of possible implementations, there is a human-intelligible part that starts at the left with the intensional extreme. A very fine-grained decomposition into many subgames is in principle possible, would still be human-intelligible, but is still not human-executable due to the multiplicity of subgames. The human-intelligible part extends at least as far as (4) in Figure 2.

The human-executable starts at the right, with the extensional extreme. By lumping together positions requiring the same move according to human-intelligible criteria, the database can be compacted. The human-executable part extends to the left at least as far as (3) and (4) in Figure 2. Thus the human-intelligible and human-executable ends of the spectrum overlap. This overlap is called by Michie the *human window*.

Michie saw the task of machine intelligence to determine for a given task whether a human window exists and, if so, to structure the implementation in such a way that it is positioned inside the human window. This to ensure that a human expert can monitor the program in operation. Thus Michie modified Turing's question of whether it is possible for a computer program's operation to *qualify as thought in the human sense* to whether the program's operation can be *followed by* human thought. The latter question is of great practical importance. One illustrative example was reported by Voysey [23], roughly as follows.

In the 1970s the Hoogovens steel company planned to automate control of their hot-strip rolling mill. Such a mill consists of a number of lines. In each of these a hot slab of steel is run through a series of rollers each of which reduces thickness until thin sheet is obtained. Successive rollers have to run at successively higher speeds; accurate control of these speeds is essential for avoiding extremely costly stoppages of the entire line.

Initially control was performed by human operators. As performance was not deemed optimal and as this seemed an ideal computer application because of the availability of accurate mathematical models, it was decided to have a mill controlled by computer. As performance was still not satisfactory, it was decided to maintain the human operators, but in a supervisory role, with the ability to override the computer system in case impending malfunction was suspected.

Such dual mode control turned out not to work in the rolling mill. The operators could not make sense of the line's behaviour when under computer control. They were tempted to intervene when in fact the automatic control system was running at its intended superior accuracy. Having learned then to stand back, the operators failed to intervene when needed [23].

Michie [13] identified the Hoogovens episode as an example of a computer application that is important enough to require human monitoring. There are many other examples, including air-traffic control, medical diagnosis, and the control of nuclear power plants. He pointed out that the example shows that human monitoring is typically not possible when the program is written with only performance in mind. If this is done, it is unlikely that human experts can make sense of the system's behaviour. They are therefore unable to exercise effective supervision. Michie called the missing ingredient "conceptual interface". As the programming of computers with such an interface was uncharted territory, Michie saw the need for a test laboratory in this area, and he chose chess endgames for this purpose.

KQKR is useful as a test case for whether a machine is exhibiting the characteristics of cognition because one can agree with Searle that no thought occurs in the operation of (1). Yet the operation of the more economical (3) or (4) involves features reminiscent of the "ideas, concepts, themes, goals" mentioned in the above quote. The fact that such features are explanatory — i.e. offer a cognitive and causal analysis of the behaviour that the machine exhibits — is a necessary condition for one to be able to say that a cognitive act is taking place.

So although we are not proposing that the presence of these specific features of chess game playing constitute cognition or thought, we do believe that, in conjunction with other case studies such as an equivalent characterization of linguistic behaviour, we would have a starting point for a more precise characterization of cognition. Indeed one would expect, from recent discoveries in the neuroscience [7], that even structurally similar abilities, such as the ability to reason with natural language and the ability to reason mathematically, would have different, albeit related, explanatory structures.

7. THE FULL GAME OF CHESS

We used the minimal-cost criterion to reject the Chinese-Room style implementation of KQKR in favour of cognitively more interesting alternatives. However, KQKR is of such low complexity that a wide variety of implementations are feasible on the cheapest computer available on the actual market. There are no longer computers on the market that are so weak that the extensional extreme is not feasible. In KQKR our minimum-cost criterion only works in an idealized market where computer systems are available in a sufficient variety of costs near zero.

It is therefore interesting to consider the computationally more demanding task of playing the entire game of chess at grandmaster strength (at an Elo rating of, say, 2700). The Chinese-Room style implementation is a database with an entry for each legal chess position. The number of these is estimated by Allis [2] at 10^{50}. Assuming 10 bytes per entry, this requires a computer memory of 10^{39} gigabytes, about twenty orders magnitude larger than the currently largest computer database system. The intensional extreme requires in the order of 10^4 bytes of code and a working memory (assuming depth-first search) that easily fits in a cell phone. The time required for the intensional extreme is many orders of magnitude greater than that for KQKR, which, in turn, is far beyond feasibility.

And yet, computer programs exist that run on present day systems and play level-2700 chess under tournament time constraints. Here our method of analysis is of more than theoretical interest: the cheapest available computers are not sufficiently powerful to run such a chess program. Furthermore, the cost of sufficiently powerful computer systems is large enough to allow a variety of speed/memory combinations at the same cost, so that we get a frontier as in Figure 5.

The universe of implementations is richly populated in principle. However, we have restricted our considerations to the von Neumann architectures that are currently available commercially, possibly enhanced by special-purpose hardware but again restricted to current chip-fabrication facilities. This universe is of course only a small part of what is possible when other types of information-processing devices are also considered, such as neural nets, whether artificial or natural. Yet it is not clear how this wider universe can be mapped in a common coordinate system. For example, how does one translate numbers of brain states, even if known, to equivalent numbers of bytes, or the slow firing in parallel of vast numbers of neurons to an equivalent number of processor cycles?

However, one need not try to fit brains and computer systems in the same coordinate system. If one is willing to drop the question "Can a computer be programmed to think?" and instead ask "Can a computer be programmed so that its operation is intelligible by humans?", then Figure 5 takes on a new significance.

Imagine a chess tournament where players are allowed to consult a computer during play. Deep Fritz is an example of a program that plays at grandmaster level. Such a program might have some limited use in alerting the grandmaster to mistakes: a mere listing of Fritz's suggestion for the best countermove could make the mistake clear to a human grandmaster.

Convincing the human grandmaster that there is a better move for him requires something else. Suppose that the grandmaster has a reasonably good move in mind, but that the computer finds a stronger alternative. Chances are that merely displaying the better move would not convince the grandmaster to change his mind. If, however, a program of Deep Fritz strength were implemented to operate in the human window, then reasons for the better move could be given, and these might convince the grandmaster to play the suggested move.

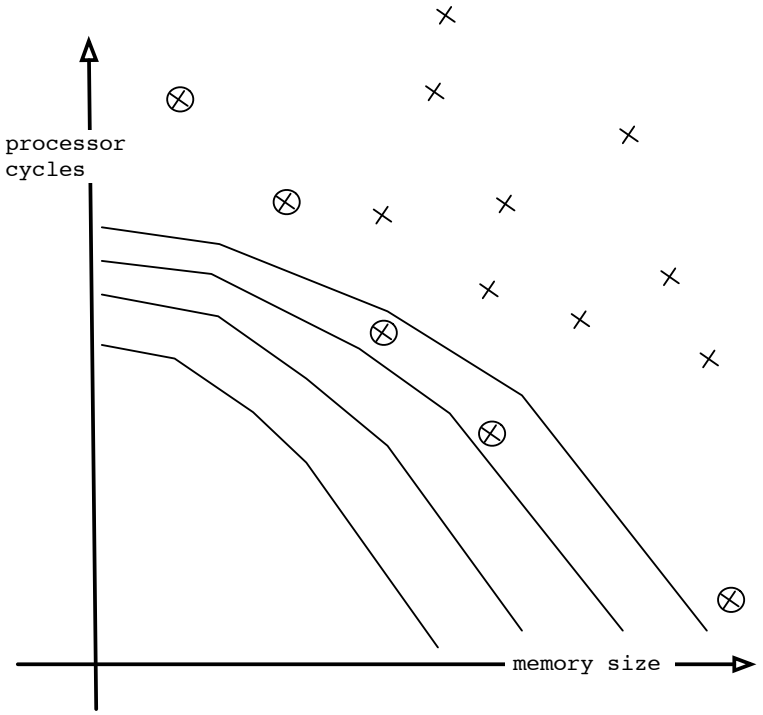

Figure 5: Trade-off frontier for the task of playing chess at rating 2700 or better. Super-imposed are lines of equal cost of computer systems. Numbers count arbitrary units of cost.

8. BACK TO THE IMITATION GAME

Although the Chinese Room Argument is usually associated with the Turing Test, Searle intended the argument as a response to Schank's proposal for scripts as a vehicle for natural-language understanding (see Appendix A). Searle implies that, from a cognitive point of view, the Chinese Room is equivalent to a system as envisaged by Schank. We can infer that Searle would, similarly, regard (3) and (4) as cognitively equivalent to the extensional extreme of KQKR.

We this we do not agree. It seems to us reasonable to believe that a "language-understanding" system à la Schank could pass the Turing Test. Implementing such a system would be a large project, one that has, as far as we know, not been attempted on the required scale. Less sophisticated programs come close, starting with Weizenbaum's ELIZA, which has been mistaken by some (admittedly naïve) test subjects for a human interlocutor. The entries in the Loebner-prize contests fail the Turing Test: they are up against sophisticated interrogators who are alerted to the situation. Still, the degree of sophistication envisaged by Schank might not be necessary to pass the Turing Test.

Turing himself may have felt that ability to simulate cocktail party chat with strangers is not a sufficient criterion, hence the passage quoted in Section 1. What is, at the moment, a valid version of the Turing Test is the ability to play chess coupled with the ability to comment on the opponent's moves and to explain own moves. Programs exist that can beat any player below grandmaster level. Yet, as far as we know, no program exists that can pass such a Turing Test, even when administered by a human chess player.

9. RELATED WORK

Block [3] described a device and used it in an argument against the validity of the Turing Test. The device is similar to the Chinese Room. Block showed the infeasibility of his device by a quantitative argument similar to the one we use to dismiss the Chinese Room.

Our technique of considering in principle the entire space of implementations generated by space-time trade-offs is relevant to the concepts of bounded rationality and bounded optimality. These concepts have made their way into the foundations of artificial intelligence via Norvig and Russell [17]. However, in our examples the rationality, though bounded, is not *restricted by* the bound: in each case, we considered the task without concessions to computational limitations. Even in the case of the full game of chess we avoided the concept of bounded rationality by retreating to the task of playing chess at a fixed Elo rating.

10. CONCLUSIONS

Thought experiments can be very powerful. They can be used to effortlessly demolish a misguided piece of work, or even better, to prevent a contemplated but misguided project to be undertaken. One virtue of a thought experiment is that it does not have to be actually carried out, so does not have to be realistic. However, as the Chinese Room argument shows, thought experiments have limitations, one of which is that a thought experiment loses its persuasive power if it is *too* unrealistic. We have shown that the Chinese Room argument by itself is misconceived when the task is restricted to the Turing Test, but that the argument is fruitful in the context of other tasks of artificial intelligence such as the chess end-game.

Michie proposed the Human Window to draw attention to the phenomenon of inscrutable software and to its danger when used in safety-critical applications. The requirement that a program's execution be both human-intelligible and human-executable offers an operational definition of intelligence and suggests that some classes of programs that solve problems for human thought can be considered intelligent if they satisfy this criterion.

Starting from the Chinese Room argument and taking into account performance limitations of computer systems we re-invented for chess Michie's concept of the human window. This suggests that human brains and computer systems are subject to sufficiently similar performance limitations and that intelligence arises as a matter of necessity when having to deal with information processing tasks of sufficient complexity.

ACKNOWLEDGEMENTS

We gratefully acknowledge Daniel Lemire and Peter Turney for their suggestions as well as Viviane Fairbank for her help with the figures. This research was supported by the University of Victoria and by the Natural Science and Engineering Research Council of Canada.

11. REFERENCES

John Preston and Mark Bishop (editors). *New Essays on Searle and Artificial Intelligence*. Oxford University Press, 2002.

L.V. Allis. *Searching for Solutions in Games and Artificial Intelligence*. PhD thesis, Computer Science Department, University of Limburg, The Netherlands, 1970.

Ned Block. Psychologism and behaviorism. *The Philosophical Review*, 90(1):5–43, 1981.

Murray Campbell. Knowledge discovery in Deep Blue. *Communications of the ACM*, 42(11):65–67, 1999.

M.R.B. Clarke. A quantitative study of King and Pawn against King. In M.R.B. Clarke, editor, *Advances in Computer Chess 1*, pages 108–115. Edinburgh University Press, 1977.

T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2001.

Roland Friedrich and Angela D. Friederici. Mathematical logic in the human brain: Syntax. *PLoS ONE*, 4(5), 2009.

Stevan Harnad. The annotation game: On Turing (1950) on computing, machinery, and intelligence. In *The Turing Test Sourcebook: Philosophical and Methodological Issues in the Quest for the Thinking Computer*, volume 44, pages 800–810, 2005.

Douglas R. Hofstadter. A coffeehouse conversation on the Turing Test. In *Metamagical Themas*, pages 492–525. Basic Books, 1985.

D. Michie. Memo functions and machine learning. *Nature*, 218:19–22, 1968.

D. Michie. An advice-taking system for computer chess. *Computer Bulletin, ser. 2*, 10:12–14, 1976.

Donald Michie. Experiments on the mechanization of game-learning, part 1: Characterization of the model and its parameters. *The Computer Journal*, 6(3):232–236, 1963.

Donald Michie. New face of AI. Technical Report 33, October 1977.

Donald Michie. Experiments on the mechanization of game-learning, part 2: Rule-based learning and the Human Window. *Computer Journal*, 25:105–113, 1982.

Donald Michie. Alan Turing and the return of the imitation game. Privately circulated document, 2004.

J.R. Quinlan. Semi-autonomous acquisition of pattern-based knowledge. In J.E. Hayes, Donald Michie, and Y-H Pao, editors, *Machine Intelligence*, volume 10, pages 159–172. Ellis Horwood Press, 1982.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, 2002.

R.C. Schank and R.P. Abelson. *Scripts, Plans, Goals, and Understanding*. Lawrence Erlbaum Press, 1977.

John Searle. Minds, brains, and programs. *Behavioral and Brain Sciences*, 3:417–457, 1980.

Claude E. Shannon. The mathematical theory of communication. In Claude E. Shannon and Warren Weaver, editors, *The Mathematical Theory of Communication*, pages 3–91. The University of Illinois Press, 1949.

Thomas Ströhlein. *Untersuchungen über kombinatorische Spiele*. PhD thesis, Technische Universität München, 1970.

A.M. Turing. Computing machinery and intelligence. *Mind*, 59:433–460, 1950.

Hedley Voysey. Problems of mingling men and machines. *New Scientist*, pages 416–17, 1977.

Joseph Weizenbaum. ELIZA – a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9:36–45, 1966.

T. Winograd. A procedural model of language understanding. In R. Schank and K. Colby, editors, *Computer Models of Thought and Language*. Freeman, 1973.

12. APPENDICES

APPENDIX A: SEARLE'S ORIGINAL FORMULATION OF THE CHINESE ROOM ARGUMENT

We have simplified and generalized the Chinese Room argument. As a result of these changes, it is incumbent on us to make it easy for the reader to consult Searle's original formulation [19], so we cite it below.

> *Let us apply this test to the Schank program with the following Gedankenexperiment. Suppose that I'm locked in a room and given a large batch of Chinese writing. Suppose furthermore (as is indeed the case) that I know no Chinese, either written or spoken, and that I'm not even confident that I could recognize Chinese writing as Chinese writing distinct from, say, Japanese writing or meaningless squiggles. To me, Chinese writing is just so many meaningless squiggles.*
> *Now suppose further that after this first batch of Chinese writing I am given a second batch of Chinese script together with a set of rules for correlating the second batch with the first batch. The rules are in English, and I understand these rules as well as any other native speaker of English. They enable me to correlate one set of formal symbols with another set of formal symbols, and all that "formal" means here is that I can identify the symbols entirely by their shapes. Now suppose also that I am given a third batch of Chinese symbols together with some instructions, again in English, that enable me to correlate elements of this third batch with the first two batches, and these rules instruct me*

how to give back certain Chinese symbols with certain sorts of shapes in response to certain sorts of shapes given me in the third batch. Unknown to me, the people who are giving me all of these symbols call the first batch "a script," they call the second batch a "story. " and they call the third batch "questions." Furthermore, they call the symbols I give them back in response to the third batch "answers to the questions." and the set of rules in English that they gave me, they call "the program."
Now just to complicate the story a little, imagine that these people also give me stories in English, which I understand, and they then ask me questions in English about these stories, and I give them back answers in English. Suppose also that after a while I get so good at following the instructions for manipulating the Chinese symbols and the programmers get so good at writing the programs that from the external point of view that is, from the point of view of somebody outside the room in which I am locked — my answers to the questions are absolutely indistinguishable from those of native Chinese speakers. Nobody just looking at my answers can tell that I don't speak a word of Chinese.
Let us also suppose that my answers to the English questions are, as they no doubt would be, indistinguishable from those of other native English speakers, for the simple reason that I am a native English speaker. From the external point of view — from the point of view of someone reading my "answers" — the answers to the Chinese questions and the English questions are equally good. But in the Chinese case, unlike the English case, I produce the answers by manipulating uninterpreted formal symbols. As far as the Chinese is concerned, I simply behave like a computer; I perform computational operations on formally specified elements. For the purposes of the Chinese, I am simply an instantiation of the computer program.