Annales UMCS Informatica AI 2 (2004) 71-80

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Mind and information processing: some considerations

Jerzy Mycka*

Institute of Mathematics, Maria Curie-Skłodowska University,
pl. M.Curie-Skłodowskiej 1, 20-031 Lublin, Poland

Abstract

The paper is devoted to the problem of relations between abilities of human mind and automatical information processing. Here we can use the notion of artificial inteligence as the abbrevation of information processing realized by mechanical devices. For the purpose of an analysis of the mentioned problem some classical notions and arguments from computer science are presented.

1. Preliminaries

The idea of mechanical solving of problems goes back to the distant past in science. We will not, however, deal with the historical outline of this issue, but immediately turn to the notions accompanying the examinations of computability which were coined by Alan Turing [1].

1.1. Turing machine

The model of computability called the Turing machine, described by means of an informal language, can be defined as follows: the machine comprises an infinite tape divided into identical cells, which is to store input data, output data and working information. All the elements on the tape are strings (sequences of symbols); at the same time there is a rule of placing one symbol into one cell. Without a loss of generality a particular alphabet is usually chosen as a range of symbols permissible to build strings. Practically the binary (zero, one) alphabet is often chosen, it allows a convenient and consistent representation of data. Moreover, the construction of the machine requires a description of a finite set of states from which an element, indicating a current situation (state) of the machine, originates.

* *E-mail address*: Jerzy.Mycka@umcs.lublin.pl

The Turing machine works in steps which are identical to the duration time. At every step the machine reads the content of a current cell (pointed out by the head), then it changes the content according to a read symbol and a present state. The same information serves successively to change the state and to move the head to the next left or right square. Particular states are distinguished as final; if the machine is in one of them, it finishes its work. The classical model of the Turing machine requires the number of symbols on the tape (different from the empty symbol) to be finite at every moment of the machine work.

The formal definition can be given as that below. A Turing machine M is the system:

$$M = \left(\Sigma, Q, q_s, q_f, \delta\right)$$

where $Q \cap \Sigma = \varnothing,\ \left|Q\right| < \infty,\ \left|\Sigma\right| < \infty$,

$q_s, q_f \in Q$,

$\delta : \Sigma \times Q \times \rightarrow \Sigma \times Q \times \{\leftarrow, -, \rightarrow\}$.

We use the following terminology: Q is a set of states of M, q_s initial state, q_f final state. Σ is an alphabet (a set of symbols) and δ a transition function.

A computation of Turing machine can be described as follows.

1. A configuration is a triple (q,i,t), where $q \in Q$, $t \in \Sigma^*$ $\left(\Sigma^* : Z \rightarrow \Sigma\right)$.
2. A computation is a sequence $\left(c_i\right)_{i=0}^{\infty}$ or $\left(c_i\right)_{i=0}^{n}$, where every c_i is a configuration and the following conditions are satisfied: $c_o = \left(q_s, 0, t_o\right)$, $c_{i+1} = \left(q_{i+1}, j_{i+1}, t_{i+1}\right)$ such that

$$\delta\left(q_i, t_i\left(j_i\right)\right) = \left(q_{i+1}, s, r\right),$$

$$j_{i+1} = \begin{cases} j_i, & r = -, \\ j_i + 1, & r = \rightarrow, \\ j_i - 1, & r = \leftarrow; \end{cases}$$

$$t_{i+1}\left(k\right) = \begin{cases} t_i\left(k\right), & k \neq j_i, \\ s, & k = j. \end{cases}$$

Moreover, if a computation is finite then $c_n = \left(q_f, i_n, t_n\right)$, and if a computation is infinite then $\left(\forall \geq 0\right)\left[c_i = \left(q_i, j_i, t_i\right) \Rightarrow q_i \neq q_f\right]$.

A problem is computable, according to A. Turing if its solution can be found due to the properly constructed Turing machine. Let us notice that the whole process of computing described above can be easily imagined as the work of a man who, by means of a sheet of paper and a pencil, realizes (thoughtlessly) consecutive changes of symbols according to strict rules. Robert Soare [2]

stressed the character of Turing's computability, naming the computing subject 'computor', to emphasize the idealization of human activity which was used here. An analysis of the machine potentials (physical systems) does not seem to be the Turing's aim. In agreement with the previous remarks the construction of Turing machines encloses rather the possibilities of 'an ideal mathematician' activity.

Of course, the Turing machine is not the only one model of computable processes. Just to illustrate, there are the Markov algorithms [3], the Church λ-calculus [4] or partially recursive functions all of which were proposed independently. In the course of research for all the models the thesis has been verified that they are identical as far as the scope of computability is concerned (compare [1]). Using an informal language: every issue explained in one of these models can be solved in any other model as well.

1.2. Mind

Human mind is a notion in relation to which there is a variety of definitions (see [5]). Here we want to propose the definition that seems to be as general as to agree with the majority of other proposals. We will introduce the following description: the mind is a human power that is used to create notions, to formulate judgements and to carry out conclusions (reasonings).[1]

Let us comment briefly on the above-mentioned domains of the mind activity (compare [6]).

Forming notions is based on placing in mind the ideas through which we express subjects of our thoughts and sensorial concretes. In a general case an appearance of notions is an effect of the abstraction process of human intellect. Notions are the source of understanding what is common for all particular units of the same kind, that can be perceived either sensorialy or through imagination.

Formulating judgements is far more complicated. The point is that notions have to be joined into sentences. The sentences are very remarkable: they always have the quality of positive or negative judgements about the presence of a particular feature (something possesses a particular characteristics). An existential judgement is very important, it states the presence of a thing (there exists the thing). Judgements always carry information about the truth or falseness of some existence.

Reasoning is a process of passing from one judgement to the other. The rules of reasoning are described by a logical account of the methods of proper concluding. The most general approach to construct such deductions is to notice in some judgements the justifications for other propositions.

[1] This definition is limited consciously to a cognitive sphere, putting volitive sphere questions aside.

A relation: mind-body is a different issue. We will not concentrate on this question as we are not interested here in a factual relation between human cognitive powers and body (and, especially, an issue of the human brain role in cognitive processes becomes irrevelant for us in this stage). A crucial element of our considerations will be limited in its efficiency to the viewpoint that a man possesses in some way cognitive abilities (the mind).

1.3. Problems of computability

Turing machines defined in the following way allow a comprehensive analysis of computability problems. The most important are: distinguishing the issues that cannot be explained by means of the Turing machine (unsolvable problems) and pointing out time and spatial restrictions originating from the nature of a solved issue (theory of complexity).

Because the model of Turing machines is universal for theory of computability, we use it as the main notion in our analysis of properties of artificial intelligence (i.e. in the analysis of potential cognitive power of computers).

Problems of decidabilty can be connected with a quantification of variables occurring in the analysed algorithms. If, for example, we can decide the logical value of some relation P for a given value of an argument x, the more difficult question is to find such value x_o that the relation P is satisfied. The above consideration leads us to the classification of computational problems by the arithmetical hierarchy.

For a function $g: N^n \to N$ its graph will be denoted as $G_g = \{(\bar{x}, y): g(\bar{x}) = y\}$. We will use an arithmetical hierarchy to classify subsets of N and natural functions by their graphs. This infinite hierarchy consists of the classes Σ_0^0, Π_0^0, …, Σ_i^0, Π_i^0, … . Each class $\Sigma_i^0, \Pi_i^0, i \geq 0$ is a family of relations (including sets) on some cartesian product of the set of natural numbers. The method of a construction is inductive: the classes $\Sigma_0^0 = \Pi_0^0$ contain recursive relations (i.e. those with characteristic functions computable by Turing machines); the class Σ_{n+1}^0 has only such elements S that the relation S is equivalent to some relation $\exists tP$, where P is in Π_n^0; the class Π_{n+1}^0 has only such elements that the relation S is equivalent to some relation $\forall tP$, where P is in Σ_n^0. By the Δ_n^0 we denote $\Sigma_n^0 \cap \Pi_n^0$. The artithmetical hierarchy satisfies the strict inclusions of its levels: $\Sigma_n^0 \subset \Pi_{n+1}^0$, $\Pi_n^0 \subset \Sigma_{n+1}^0$, $n \geq 0$.

The importance of the arithmetical hierarchy is connected with many fields. It can be viewed as a way of a formal description of definiability problems (see [1]). Its classes can be used to classify ‘a complexity’ of mathematical notions

(e.g. the definition of a limit of sequences is of Π_3^0 class). From the point of computability theory we can see the arithmetical hierarchy as the levels of natural functions (given by their graphs), which are different in quantity of infinite 'while' loops necessary to their computation. Also linguistic problems of computer science can be expressed in term of this hierarchy. The most known example are classes of recursive $\left(\Sigma_0^0\right)$ and recursive enumerable $\left(\Sigma_1^0\right)$ languages.

It is worth observing that many important problems as the halting problem or functions identity problem are strictly in the class Σ_1^0.

Let us add a few words about nondeterminism. The version of Turing machine described above is deterministic. This means that for given input data there exists only one possible computation. However, we can modify the definition of Turing machines in the following way. For nondeterministic Turing machines the next action described by a triple: (a state, a symbol and a move) for some combinations of contents of the current symbol and current state need not be unique. The result is reached by any sequence of moves, which leads to the final state. Technically speaking, we should replace a transition function by a transition relation. In this case a computation has to be given by a tree (not a sequence).

2. Decidability

In this part we will concentrate on an analysis of unsolvability issues. Let us start from recalling the halting problem. The aim is to construct the Turing machine *H*, which adopting a coded description of any Turing machine *P* with a set of data *D*, produces *YES* signal if *P* stops for *D* data, or *NO* signal as a result of the infinite computation of *P* for *D*.

We are going to introduce certain changes in the machine. It is easy to rearrange it so that instead of *YES* signal an infinite loop of steps is extorted. One more modification can be proposed: instead of a code of the machine description and data we will supply only the code, while the program multiplies this code and adds it as data. Such a modification of the *H* machine will be called *HMOD*. Let us analyse *HMOD* operations. If the *HMOD* machine for the data which are *HMOD* code, did not finish, *H* for the same data would produce *NO* and, consequently, *HMOD* would stop operating with *NO* result. In this way we obtained a contradiction: *HMOD* stops when *HMOD* does not stop. A similiar reasoning shows a contradiction appearing on the assumption that *HMOD* finishes its operations. The above argument leads to an important conclusion: the Turing machine, controlling halting of arbitrary given Turing machines with fixed data, does not exist. The result is technically described as the unsolvability (or undecidability) of the halting problem. Using the previously

mentioned arithmetical hierarchy the halting problem can be classified as an element of the Σ_1^0 class. A lot of other problems appear in a similiar situation (e.g. the identity of programs, Post's problem of words correspondence).

However, it seems that from a human point of view the situation is totally different. There is a strong conviction that an experienced programmer is able to solve the halting problem for any program. Roger Penrose [7] suggests that the difference appears in a different kind of the human mind activity. Namely, every mind operates nonalgorithmically, referring to a direct insight into questions and using an original approach in the places that usual algorithm does not work (cannot be applied).

3. Productivity

In this section we are interested in the productive sets. The definition of this notion can be given in the following way.

A set A of natural numbers is called a productive set, if there exists such a function f, that f is computable by some Turing machine (f can be partial) and for every number n we have: if the set of arguments of the n-th[2] Turing machine for which it stops is a subset of A, then $f(n)$ is defined, $f(n)$ is from A, the n-th Turing machine for $f(n)$ gives an infinite computation.

Informally speaking, a set A is productive, if it is impossible to find an algorithm which decides whether some argument is from A and, moreover, for every proposed algorithm we can effectively point out a counterexample. In this case a counterexample is such an element, that it is in A but it cannot be accepted by the proposed algorithm.

A problem whether an element is from a productive set is undecidable. And we can prove that it is a harder problem then the halting problem. If we observe the arithmetical hierarchy we can find that productive sets are neither in the class Σ_0^0 nor in the class Σ_1^0.

Let us explain the issue of productivity using an example. Selmer Bringsjord and Dave Ferruci considered in their paper [8] the problem of productive sets on the basis of recognizing interesting and well-planned works of fiction. For a change we can also have a look at films. It is not difficult to point out well-constructed and interesting films, as well as these which do not have the qualities. But an attempt to prepare an algorithm deciding about it requires specifying some criteria. Any features of a film will be taken into account on this list (dynamics of action, art of photography, conspicuous characters, good music, et al.) it is always possible to give an example of a real masterpiece, which, however, does not satisfy these criteria. In such an aspect a film collection can exemplify a productive set, similarily to library of written works.

[2] In some given enumeration.

In this connection a conclusion appears that a problem of film selection is nonalgorithmic to a degree much exceeding the halting problem. However, an everyday practice shows that choosing good works by people is relatively easy. There seems to be a significant agreement in the estimation of films, though, as it comes from a productivity description, it cannot result from an algorithmic selection of the required qualities list. As it may be observed the human mind can be equipped with specific noncomputable features of estimation that allow solving productive sets problems.

4. Meaning

The fragment will be devoted to examining the problem of the meaning connected with human and machine acting. The clasical philosophical approach links meaning with the theory of sign. There are two types of signs distinguished (cf. [9]): instrumental signs and formal ones. Instrumental signs both point out another object and they themselves are independent objects accessible in recognizing. In the case of formal signs there is no possibility of their independent approach – they always remain dependent on the function of denoting the signed objects. That is why the meaning in human cognition is, in this concept, connected with possessing, by the human mind, ideas which are formal signs. They become as transparent references that cannot be considered in their essence but relate us to the objects they refer to and give meanings.

In this context let us mention a well-known mental experiment of John Searle called a Chinese room. Let us consider the room into which some stories written in Chinese are passed through special openings. A man inside, who does not know the language, referring merely to the outlook of the symbols acts according to the given instructions. They order him to prepare the next set of Chinese symbols that are passed outside the room afterwards. With a perfectly prepared set of instructions an external observer is a witness of a dialogue in Chinese. The fact of an exchange of sentences in the Chinese language might prove that the man in the room understands the language. However, knowing the rules of the experiment we know that the man is not able to join any meaning with the operations performed.

The above description can be regarded as an exemplification of differentiating the syntactic and semantic levels in data processing. Computers are limited to processing sequences of symbols without any reference to their meaning. A man during a cognitive perception is able to use the semantic (meaningful) content of the received information.

The issue is connected with Gödel's first theorem [10]. Here it will be presented in the following form: there is no consistent, complete, axiomatizable theory which is an extension of Peano arithmetic. This means that such a theory includes some undecidable sentences. In the proof of this theorem the sentence that codes specific information is used: "I (the sentence) have got no proof".

Though formally (syntactically) the usage of the proof technics allows neither to prove nor to refute the sentence, the meaning of this fact is that the sentence can be regarded as true.

Once more it would suggest the superiority of the semantical level and the abilities of the human mind to use meanings in drawing conclusions that exceed machine potentials.

5. Determinism

In the final part we will deal with some problems of determinism and nondeterminism. Let us recall the fact that a deterministic character of the Turing machine computations is connected with a complete designation of a computation on the basis of information about input data as well as the machine construction.

The law of casuality plays the role here, which guarantees that each step of the Turing machine operations will take place with complete necessity. It seems that the choice range of the way a man and a machine function may become completely exhausted by adding nondeterminism as an alternative method of procedures. In this case at least in some moments, there is a situation in which maintaining the system – though limited to a couple of variants - is completely nondetermined. In other words, casuality does not work any longer and a coincidence comes into prominence. The coincidence means here a lack of reason for choosing one of possible solutions, not a phenomenon that cannot be foreseen.

Without solving the question whether physics allows admitting the construction of nondeterministic devices, let us notice that both the cases of operating are totally explainable. In the case of determinism we can describe a full sequence of causes leading to an achieved effect. In the event of nondeterminism, however, it is sufficient to add the results of accidential choices leading to the considered situation.

Let us focus on one of the standard questions of artificial intelligence. A test of behaviour is sometimes proposed as the measure of achieving the level of human intelligence by machines. Alan Turing [11] is the author of this idea, whose essence is based on investigating the pair: man – machine in a way that does not allow to observe directly the author of answers. If, statistically, it is impossible to distinguish a man from a computer, one can claim that the computer reached the level of human intelligence.

It is worth noticing that such a test has a merely behavioural meaning. Functioning of a computer is not crucial here, we estimate its potentials in a completely operational way. Hence the Turing test, even if it succeeds to the highest degree, proves only that a mechanism has been constructed, which can pretend entirely to be a man. This restriction can be found in Ada Lovelace's works [12] who points out that anything computing machines will be able to do,

will always be a realization of the program that was assigned to the machine by its creator.

Yet, such a way of operating is completely insufficient in a description of human activity. One of the characteristic features of human nature is autodeterminism. It should be clearly underlined that at least in some aspects of man's activity his behaviour is neither determined nor coincidental. This fact is commonly approved as people in general admit a man's responsibility for his deeds. A notion of responsibility would be pointless if a man were regarded either as a deterministic or nondeterministic mechanism.

On account of this, the Lovelace test can be chosen as means to estimate the real level of approximation of computers to human abilities. In this case the point is to establish whether the creator of the machine is able to explain its operations, possessing all the information about its resources. Where such an explanation cannot be given, independent and creative characteristic of the machine would be reliably displayed.

6. Conclusions

All the above arguments are not proofs in the sense approved by formal sciences (mathematics, logic). The whole reasoning, in fact, is based on the meeting of two different worlds. One side is represented by computer science by means of notions of the computability theory. That is why we have here complete precision of reasoning and a possibilty of using mathematical tools. The other side comprises the issues of a description of human being, his activity and cognitive power. This domain belongs partially to psychology, but exceeding the behavioural sphere, it has to make allowances for philosophical anthropology.

Thus, the situation seems to resemble a problem of the Church-Turing thesis. Namely, it is impossible here, because of the difference between the characters of compared beings (the mind versus a computer model) to enclose the question in the boundaries of computer science. It does not mean, however, that because of it, the issue becomes irrational. It is necessary to use here proper methods.

It seems that a level appriopriate to compare the mind and a computer is the level of philosophical reflection. Of course, it must take into account the results obtained by computer science. However, we cannot avoid accepting certain solutions within the sphere of philosophy.

The proper domains of philosophy that need considering in the above-type analyses are epistemology (what are the kinds and possibilities of limiting cognition) and anthropology (what is a structure of human being, his mind and cognitive powers). In this paper the author tried to use the achievements of peripatetic and Thomist philosophy to refer to these domains. The questions about these analyses, being carried out on the basis of different philosophical options, remain open for further research.

References

[1] Odifreddi P., *Classical Recursion Theory*, North-Holland, (1989).
[2] Soare R., The history and concept of computability. In: E. Griffor, (ed), *Handbook of Computability Theory*, Elsevier, (1999).
[3] Markow A. A., *Trudy Matiematiczeskogo Instituta Stieklova*, Theory of algorithms, 38, (1951), 176, (in Russian).
[4] Barendrengt H. P., *The Lambda Calculus*, North-Holland, (1981).
[5] *The Oxford Companion to the Mind*, Oxford Press, (1998).
[6] Świeżawski S., *Saint Thomas reread*, W drodze, (1995), in Polish.
[7] Penrose R., *The Emperor's New Mind*, PWN, (1995), in Polish.
[8] Bringsjord S., Ferrucci D., *Artificial Intelligence and Literary Creativity: Inside the Mind of Brutus, a Storytelling Machine*, Lawrence Erlbaum Assoc Inc, 1999.
[9] Adler M. J., *Ten Philosophical Mistakes*, Medium, (1985), in Polish.
[10] Gödel K., *Collected Works*, Oxford Press, I (1986).
[11] Turing A., Computing machinery and intelligence. *Mind*, 59 (1950) 236.
[12] Bringsjord S., Bello P., Ferrucci D., Creavity, the Turing test and the (better) Lovelace test. *Minds and Machines*, 11(1) (2001) 3.