Evžen Kindler

Can cybernetics contribute to the study of computer development?

Kybernetika, Vol. 9 (1973), No. 6, (475)--482

Persistent URL: http://dml.cz/dmlcz/125837

KYBERNETIKA — VOLUME 9 (1973), NUMBER 6

Can Cybernetics Contribute to the Study of Computer Development?

EVŽEN KINDLER

The paer contains some critical notes about the conception of development of computers, which ressembles conceptions of scholar history but not cybernetics. An experiment how to cybernetize the research of the computer development is presented: it concerns the development of the relation between data and programs in which certain factor exists that external differences between data and program tend to disappear.

1. INTRODUCTION

1.1. Computers as subject of history

The subject of the present paper is the development of computers as a subject of cybernetics. Of course, both the terms of that subject represent two distinguished types of typical subjects of cybernetics: we need not emphasize that *computers* have been studied by methods of cybernetics, and *development* as well (usually by other methods of the same science). But if we put the grammatical relation of determination between them — obtaining a new concept, the *development of computers* — both groups have no importance: what one has used for the research of development generally gives no important results for the development of computers, and what cybernetics said about computers does not concern their development: it concerns only computers as static objects, neglecting their laws of change from type to type in course of time.

The description of computer development is often presented at meetings of serious cyberneticians but they accept such a description, which is as exact as similar descriptions existing in scientifical branches which had great problems with exact expressing of their information. If we hear that the older computers had certain technological parameters, a certain operation speed, certain software and a certain contact with their neighbourhood, while the more modern computers have those aspects modified,

it is similar, as when we heard in the grammar school, that in a certain century a revolution in a certain country has removed a king, that in the same century a volcano has destroyed a town, while in the following century a famous composer lived. Of course, we can meet a result that certain relation, e.g., between a technological parameter and a software is formulated, but it is never a general aspect which could be valid for the computer development in any phase.

Together with the history of computers, their prognostics exists: it is a prolongation of the history into time which has not existed. Similarly as in the history which does not concern the computers, the history of computers becomes science fiction if it is prolongated to the future: some properties are transferred without modification, some are extrapolated in a random way, and some are neglected and, often without reason, replaced by some other properties.

Thus a problem arises: do there exist properties which can be considered as absolute and typical for the development of computers? In the following text, some properties are studied, which seem to be of that type. Some of them are found not to be suitable; however, one particular property is shown to be typical for every phase of the computer development.

1.2. Mathematical methods

As cybernetics has tried to be an exact science, the use of mathematical methods offers to be a tool also for expressing the laws of the computer development. Nevertheless, only very special aspects of it have been reflected by mathematical means, and thus they have no importance for expressing any general rule, valid in any phase: their limitations are equivalent to their relativity, because we cannot be sure how long their terms, though related by exact relations till today, will have their meaning. We know a lot of graphs, approximated very well by linear or exponential relation, where one axis represents the time of the computer development and the other represents some parameter as the cost of an operation, operation speed, capacity of certain type of memory, number of computers of certain type etc; although we can extrapolate the graph to the future, we cannot tell nothing about the interpretation of the value, reflected by the second axis. For instance, one predicts the relation between the numbers of data processing computers and real time control computers according to the geometrical properties of the graphs concerning both of them, but nobody knows if in the prognosed time any difference between both types will exist.

The author can present a suitable example of such sophistics that he heard some years ago: his colleague made graphs of the development of the speeds in the control and arithmetic units of the computers and compared them with the graphs of the development of the speeds of external memories. From certain geometrical properties it was clear that there will be a moment when the disproportion between the electronic units and the auxiliary memories will be so large, that when compiling, e.g., from ALGOL 60, we shall not need to have prepared subroutines in the machine code: they will be also in ALGOL 60, because in such form they would occupy a small space in the

auxiliary memory and they would be read and translated in a shorter time than if they were read without translation from a larger space of the auxiliary memory. Of course, that consideration seems to be very exact and effective but one might not forget another phenomenon, which exists out of the properties which are reflected by the mentioned exact consideration: similar disproportions in the speeds of the electronic units and auxiliary memories have greater importance in other fields than in automatic programming; it is for instance data processing, where unfortunately the disproportions cannot be eliminated as the automatic programming theoretically has offered. So the whole problem needs another solution. Nowadays we have recognized it very well: it is multiprogramming which has caused that the result of the exact consideration mentioned above is completely wrong.

Let us note that such methods cannot be considered as cybernetical: they are very primitive though they are exact and one can compare them with classical graphical methods used even in biomedical sciences many years before cybernetics.

1.3. Computer generations

The idea of the generation, introduced into the computer development, seemed to be a suitable implement. It has eliminated a lot of misunderstanding, forming suitable cathegories in the description of computers which have been designed. But from the general viewpoint it has no efficient meaning, too: under the concept of a generation, certain phases in the computer development appeared to be static, but one cannot do an extrapolation from one generation to the following one; let us mention the access to the computers: the familiar one, existing in the first generation, has been replaced by another type of access in the second generation, which seemed to be more modern and which applied the great speed of computers so that the user could not be directly at the computer console. But the third generation of computers has brought a negation of negation in form of terminals that have enabled to restitute the familiar access to the computer without destroying the advantages obtained by the second generation. Similar affair has been in the size of computers: smaller computers of the first generation have been replaced by larger ones in the following generation and during the phase of the third generation computers (we can say in the middle of it), minicomputers become to be very important. In other aspects, however, the development did not respect the hegelian law of the negation of negation.

The concept of computer generation is suitable to be a basis for a certain time scale by means we can simply determine events in the computer development, but it does not explain it. It can be also a suitable basis for a simple description of a computer regarding to its function and place in the general development of computers (we know, for example, computers, designed and/or manufactured in the phase of the third generation, which are facilitated by software typical for the first generation.). The popularization of the computers by their manufactures uses also the term of generation very freely and so we can see that the length of a generation time in the computer development becomes shorter and shorter and some manufacturers often create a new generation (in words) rather than a new type of computer. Originally

the concept of computer generation had a greater contents than now and we can hope that the manufacturers of the computers will offer another concept to the scientists, which will have also a great contents and a great interest, at least in its initial phase. The concept of generation can be also compared to the historical concepts of ancient age, middle age and modern age, which are also a basis for a certain time scale in certain social sciences; and similarly with the prognostics in history we can see that speaking on the future generation of computers ressembles more science fiction than cybernetics.

2. DATA AND PROGRAM

One can therefore conclude that there is no possibility of a serious formulation of any property of the computer development, because πάντα ῥεῖ. That can be admitted if we limit our consideration only to material properties modified during the existence of computers. But we can base our considerations on facts which are related directly to the computer essence. Such considerations are joined with the properties which characterize the computers and which distinguish them from the other sorts of things. In the following parts we shall observe such a property: it is the relation between program and data, about which we can observe that during the development of computers it will be modified, but in certain sense: the real difference between program and data can be considered as existing in any time because it is suitable for aplication of computers as realizators of functions (or transformers of data of certain general form in certain way which is independent on those data). But the formal and physical properties which reflect that difference will be smaller and smaller because the other expression and thinking means of the humans do not need such a difference and the human wants to make his contact with computers similar to his contact with other systems, where he does not distinguish between data and programs.

Thus we can formulate one aspect of the computer development which we can consider as a general one, valid always in the development of computers: *the formal differences between data and programs are and will be diminished but they will never dissappear*; if any partial difference dissappears there will rise another property satisfied in different ways for data and for programs which will have importance in the following development of computers as the property the satisfying of which will become equal during the next phase. Let us observe this idea in a greater detail.

2.1. Theorems of mathematical logic

It was already before the age of computers when the theorems was formulated and prooved which can be transformed into the physical reality of the computers so that instead of sets of programs a "universal" program can be made which runs like any program of the set if the data are enriched by a certain parameter. Thus mathematical

logic has told us about the possibility how to transform programs to data. Of course, it does not speak of the psychological and social aspects of that transformation.

The theorems are, e.g. about the existence of the universal normal Markov algorithm (see [1], part IV., par. 2, theorem 1.1), the universal Turing machine (see [2], [3], [4]), about the explicite form of recursive function (see e.g. [5], part 17, par. 10, page 139 or part 18, par. 7, page 148, or [6], part III., chapter XI., par. 58, theorem IX, or [7]).

2.2. Programs and data in computer hardware

The material synthesis of program and data properties in the computer memory has been realized very soon in the computer development by the idea of automatic computer of Von Neumann, who has designed a system of information processing with a unique memory for both data and program. Thus programs can modify their own instructions and data can be used as instructions of programs. There is a certain difference between the interpretation of the instructions read from the memory during the computation and of the processed data. The difference can be logically removed so that we consider the computer as a special pourpose one which can realize only one program, wired inside. This program reads the data corresponding to the instructions which are interpreted as switching to various computer operations over the other data. This conception, however, neglects the psychological aspects of programming; further material diminishing of the distinction between program and data can be expected in the nearest future by implementing the possibility that one can control parts of computer units, or modify standart operations of the computer by means of data (microprogramming is an example of such a possibility).

2.3. Programs and data in computer software

The idea of Von Neumann, mentioned in the preceding paragraph, has been reflected in the operating systems of computers so that the input file makes no substantial difference between the records of programs and those of data: series of such records are only preceded by control records which are also similar in both cases, whether they precede data or program; they differ only in certain "parameters" which switches the operating system to perform special action, often different for data and for programs.

There is no reflection of Von Neumann's idea in classical programming languages as ALGOL 60 or FORTRAN: they distinguish essentially between the program itself and data, and there is no possibility that a program could change its proper instructions (statements). Von Neumann's idea has been used only in the translators and in the interpreting programs so that they process the source program as data and generate eventual target program to which use they to jump. This aspect is, however, similar to that, mentioned at the beginning of the present paragraph in relation to the operating systems.

The phenomenon that certain different properties of programs and data are removed, exists in the programming languages only in the third generation phase: it is realized in two main types; in the *conversation languages* certain differences between data and programs do not exist because the user gives the demands to the computer in a form where the data are incorporated into the program; in the *general pourpose third generation languages* as SIMULA 67 or ALGOL 68, the familiar relation between data and programs is realized, so that both of them can have attributes of structures which are handled as "individuals" equivalently in case that they have only program attributes, or only data attributes, or both of them.

2.4. Prognoses

We have learned from the living systems that it is not necessary to distinguish between data and programs from the viewpoint of any external property. If we want to cause an animal or a human to do some action we give him signals where the character of the action is coded together with its parameters. Though we know that we can assume another – more general – program in his neural system, for which our demands form data (we can say metaphorically that the demands are not distinguished to programs and data for that more general program similarly as the input file is not distinguished into programs and data for the operating systems, or similarly as the data and programs are not distinguished in a conversation with a computer facilitated by a conversational system of automatic programming), we usually neglect it in many usual situations. If we want to communicate with a computer as with a specialized but powerful collaborator we shall realize systems of control for computers similar to those which exist for living beings. So the users of computers will be more and more in situations where less and less differences between the form of program and data will exist.

On the other hand, preparation of programs which perform certain actions with any data of certain very general classes is a result, which has been accessed by the computer development till now and the value of which is rather important so that one cannot assume that it might be neglected in the further development, even if the form of implementation can be different. Thus we can expect that certain typical and essential properties of programs will distinguish them always from data, but we cannot expect that they would be interpreted in physical form of signals given to the computers.

Let us note that we could have an impression that the difference between the programs and data is completely subjective, reflecting only the intentions of the user: what he takes as a function should be program and what he takes as its arguments should be data. The author would like to refer to the principle of ancient philosophers that the laws of meaning reflect the laws of being: thus it would be certainly something objective in the difference between data; but we cannot express it because there is no

suitable ontology of computers and their software (such an onthology would be create in the nearest future also for many other reasons!).

3. CONCLUSIONS

We have presented a property which seems to be one of the general aspects of the development of computers, having its proper meaning always during the development of computers. We do not state that it would be the unique property which characterizes the development of computers and that there would not be any other property of such qualities, which would tell another information than the presented one. The author hopes that there will be soon discovered other properties valid always in the development of computers and able to be studied by methods typical for cybernetics, but he feels not to be able to express them technically in the present paper (experts in probability can surely formulate other properties related to their interests, specialists in system theory can do the same in their field etc.).

3.1. Hardware and software

Newertheless, the author would like to propose to pay attention to a property which seems to be in certain sense a special case of the property studied in the preceding part, but in the physical interpretation it can give more information. The relations between data and programs can be simply transformed between software and hardware: let us consider computer as a specialized single device which can read data representing instructions and interpret them as one "address" of the instruction, equivalent (in logical meaning) to the other address representing the "efficient" data; then we can make the considerations mentioned above also for such a special program (wired in it). They will be similar to the considerations, noted at the end of 2.2. Of course, a greater development of such considerations would be platonic present days because it has no realization: the difference between hardware and software is clear enough and constant. But it has been caused that nowadays the transport of energy and that of matter is clear enough in the computer, too, because the transport of matter can be simply neglected in it. When the transport of matter in computers will be more important (when the computers will repare and modify its own hardware) we can expect the processes where we cannot state if the main factor which carries the information is matter or energy and so certain properties which distinguish hardware from software will be removed, because hardware is an affair of substance while software is represented at computers by energetic signals. We can state that this aspect of the computer development exists but only in its initial, singular state. When it leaves it we shall be able to study it in a more interesting way; nowadays we can have use only of the analogies between the hypothetical situation in future and the present state in the living systems where the familiar contact between hardware and software has been already realized (in genetical information, neural system, etc.).

The property presented in the second part of this paper can serve as a very clear illustration of the difference between mathematics and cybernetics. If we are limitted to mathematical objects only, we can finish any discussion on that property if we have proved mathematical theorems mentioned in 2.1. But if we are cyberneticians we must study that property from all viewpoints, namely concerning its relation to psychology, social environment, technology etc. Some of them can be expressed in mathematical terms and studied by mathematical methods but the science of cybernetics must determine problems before such an abstraction.

Acknowledgements. The author would like to express his thanks to Czechoslovak Association for Cybernetics of the Czechoslovak Academy of Sciences, which has enabled him to discuss the presented ideas with a very wide forum of specialists, of which namely Prof. Dr. K. Čulík, DrSc. and Doc. Dr. J. Habr have presented good stimuli to make considerations more exact.

(Received March 8, 1973.)

REFERENCES

[1] А. А. Марков: Теория алгорифмов (Theory of algorithms). Труды математического института имени В. А. Стеклова, XL-II, Издателство Академии наук СССР, Москва—Ленинград 1954.

[2] A. M. Turing: Computable numbers with an application to Entscheidungsproblem. Proc. London. Math. Soc. ser. 2., *42* (1936—7), 230—265.

[3] C. E. Shannon: Universal Turing machine with two inner states. In: Automata studies. Princeton University Press, Princeton 1956.

[4] M. D. Davis: Note on universal Turing machines. In: Automata studies. Princeton University Press, Princeton 1956.

[5] R. Péter: Recursive Funktionen. Akademiai Kiadó, Budapest 1951.

[6] S. C. Kleene: Introduction to Metamathematics. D. Van Nostrand Comp., New York—Toronto 1952.

[7] S. C. Kleene: General recursive functions of natural numbers. Math. Ann. *112* (1936), 727—742.

PhDr. RNDr. Evžen Kindler, CSc., Biofysikální ústav FVL KU (Biophysical Institute, Charles University), Salmovská 3, 120 00 Praha 2. Czechoslovakia.