Acta Cybernetica **18** (2007) 9–14.

The activities of László Kalmár in the world of information technology

Árpád Makay*

Abstract

Since the end of the 1950s László Kalmár has been interested in the information technology. During a 20 years period he designed several variants of computers interpreting high-level programming languages on architectural levels.

Keywords: logical machine, formula-driven computer, high-level language interpreter machine

Towards the end of the 1950s, information technology (IT) became one of the fields in which László Kalmár was highly interested. He was clearly aware of the rapid spread of computers and their excellent applicability for numeral computations. It is nevertheless extremely likely that the links between mathematics and IT were what caught his interest and shaped his views of this field. It is undoubted that he tackled problems from the aspect of a mathematician, always attempting to apply mathematical methods in a world, which at that time was virtually purely technical and technological. It soon became obvious that IT requires and makes wide use of the laws and methods of mathematics: it may suffice merely to mention the inspiring role of automata theory or coding theory. The exactness of mathematics is reflected in the problem solving of IT, the precise understanding of the problems and their detailed analysis, which often demands considerable work. Kalmár's interdisciplinary knowledge played a significant role in his continuous search for new areas of use of IT, defining concrete problems for which he often found solutions and attracted the interest of researchers and developers.

It should be remembered that the freedom of researchers to carry out effective work in Hungary in that period was restricted by a number of factors. The technical resources in the country were rather poor. For understandable reasons, most of the resources were placed in the service of the economy and the running of the state, only a minor part being made available for teaching and research. Kalmár's wide-ranging contacts and (from the 1960s) his nationwide recognition in the world of IT helped him overcome many of the technical obstacles, especially

*University of Szeged, Árpád tér 2, Szeged, Hungary.

in the practical teaching area. Nonetheless, the need to adapt to the external constraints certainly influenced his thoughts and plans. One of the main areas of his interest, the combination programming languages and mathematics, and especially the formal language of logic, was restricted from the outset, the limitations being set by his ambition for the availability of the necessary resources.

Kalmár was unceasingly convinced that the already considerable development of the IT world could benefit still further from the innovative work of Hungarian researchers. In consequence of his international reputation, the leading journals and technical books were at his disposal, often as complimentary copies. The world was generally open to him. At conferences, he was able to meet internationally respected researchers and to set out his ideas and achievements. In this way he acquired up-to-date information on the areas of perspective research and development, which he readily shared with his students and colleagues.

When an effective tool such as the computer becomes available to an individual, his or her imagination suddenly catches wings. And this is what happened to the early IT researchers, engineers and end-users in Hungary, among them László Kalmár. As an example, the machine translation of natural languages seemed attainable from the very beginning. Kalmár closely watched and supported the Hungarian group working on this project. We now know that this goal was reached in part only much later. For that group at that time, the objective appeared unattainable, though their activities furnished important information and knowledge relating to the field of linguistics.

László Kalmár was no stranger to philosophy. Perhaps this was one of the reasons why he became interested in some of the unanswered questions of mathematical logic which (with full mathematical exactness) touched on the limits of reliability of mathematics. Questions often arose in IT (also referred to as cybernetics) such as those concerning the relationship of man and a "thinking" machine, the ability of a machine to reproduce itself, and the controllability of computers. Kalmár developed his own concepts of these issues and often put forward his ideas at appropriate forums. He did this mainly as a mathematician, a stranger to exaggeration and science fiction.

László Kalmár worked in the purely theoretical realm of mathematical logic; it may be stated that he was a real theoretical researcher. In the world of IT, however, he strived towards concrete instruments. The technology was limited, but he designed tools that could be constructed.

The first project that was achieved was a by-product of a departmental seminar. The theme was the technical implementation of mathematical logic (propositional logic). This is an exciting topic if it is considered that the active parts of computers are logical circuits, the tasks of which are logical calculations. The decision was taken to build a "logical machine" that computes the values of logical formulae [1].

The formula applied could contain a maximum of 8 variables, and all the basic operators of calculus could be used. A double contact switch represented the value of a variable or a component formula. One pole had the value TRUE, and the other one the value FALSE. Accordingly, an operator of two variables needed two input and one output switches. Special cables connected the poles. These connections had to be set according to the logical operators; it may be said that the operators were programmed. The input and output poles of the boxes, already programmed for the basic operators, were appropriately connected to each other so that formulae of desired length became computable. Operator priority and the use of parentheses were enabled. It is obvious that the machine did not tackle the problem by reducing the formula to some kind of normal form. The values of the maximum 8 operands were set by a simple series of switches, their values and that of the formula being indicated by lamps on the display.

The logical machine was a demonstrative tool, but it led to a degree of self-confidence necessary for more difficult challenges to be tackled. Every creator feels the need to explain what his or her creation is good for: the formula built up from the boxes is a circuit that is being tested by the machine. This activity was supported by a relayed "memory", together with the potential for the simultaneous handling of a number of formulae. By 1959 the machine had received a new application (one necessary in most computers): it had become a binary adder.

It was roughly at this time that the training of "programming mathematicians" started at the University of Szeged, and these courses became increasingly more popular. The characteristic features of the courses were very thorough training in mathematics and programming, first in assembly, and later in higher-level languages. While providing several mathematical courses, László Kalmár was the professor of machine programming.

In the 1960s, he experienced that programs written in assembly (or in direct code) were more effective than the codes generated by the compilers, not to mention the time and memory requirements of the compilation process. The effectiveness was a result of the work of the programmer in searching for the memory and time optimum. He also observed that the programming work, i.e. the human energy invested in problem solving, is more effective if a higher-level language is used. This latter is nowadays held to be of greatest importance. At that time, Kalmár could not predict that within 25 years the memory and computing capacity of computers would have become virtually limitless, and that the abilities of compilers would have been enhanced considerably as a consequence of theoretical results. He believed that the solution lay in the approach of machine code to the syntax and semantics of higher-level languages.

During his productive IT activities, Kalmár often returned to this idea of a formula-driven machine. The possibilities available in the various periods are reflected by some of the versions planned throughout the years. In parallel, his goal was a definition of the computer as an algebraic structure, his plans being constructed on this precise theory. As an example, he looked upon a computer operation as (amongst others) a transformation in the memory state. However, because of the large number and complexity of the operations, the characteristics of this transformation could not be written with mathematical exactness. In order for this to be done, the model should have been brought into a much simpler level, e.g. to the level of Turing machine theory, but the practical demands did not allow this. Accordingly, the algebraic model rather played the role of a general approach and was not used directly in the design.

The first plans involved the use of the Ljapunov operational language interpreter machine [2]. The language allowed the use of a limited number of variables, expressions built up with the applications of arithmetical operations, and a few algorithmic tools: conditional clauses and cycles. The syntax was straightforward. Lexical analysis was barely needed. Because of the lack of block structures and program segmentation, there was no need for the most difficult techniques applied in modern interpreters, which at that time were probably impossible to implement with the technical support available then.

The computational unit of the machine was a stack-like structure built up from register quartets. These were designed to handle the arithmetical and logical operations of two operands; they stored the two operands, the operation and (once the values of the operands were available) the result. The result register of the register quartet was connected to both operand registers of the higher-level register quartet through gates. At most one of the gates was open, in order to receive the result of the operation computed at the lower level.

The program was run through the sequential reading of the characters in one pass. At all times there was one active register quartet and one of its registers was active. In one step, the value (if any) of the next variable from the sequence was placed in the active register. In every step, the state of activation of the current register quartet and its register was refreshed, and the states of the gates were set. If the operation could be computed (both operands present in a register quartet), the operation was performed and the result flowed upwards through the open gate.

An analysis of the system reveals that simple cycles can be implemented with the described register hierarchy, since the repeating condition is also an expression. Apart from this, a control unit was needed, with the role of interpretation of the sequential, conditional and cyclic clauses. The memory assignments and the indexed variable handling demanded a special design. Kalmár's plans included all of these features, but the result of prime importance was the technically applicable, special stack architecture. The conditions for the building of the machine could not be met within Hungary, but parts of his machine plans were utilized in the MIR machine of the Ukrainian Academy of Sciences, built in 1966.

By the beginning of the 1960s, it was evident that the stack was an extremely powerful tool in IT. If the traditional infix expressions were converted to postfix form and put into the stack, their interpretation was child's play. The use of a stack simplified the conversion too. If the algorithmic parts of programming languages such as ALGOL-60 could be converted to postfix form, then (by means of a one-pass compilation) the program could be run several times without being compiled in the classical machine language. This was more or less achieved, and extremely efficient interpreters were developed on the basis of this theory. The efficiency was further increased by building the stack into the hardware level of the architecture of the computer, together with the technique of microprogramming.

In the meantime, theoretical results were obtained that gave a definitive direction to the evolution of IT. The design of efficient lexical analysis algorithms was based on the theory of finite automata. The theory of pushdown automata and their special classes defined the limits to be taken into consideration within the syntax of programming languages for the building of efficient compilers and interpreters. The limitations were loose, and the requirements of the programmers concerning the programming languages could therefore be fully satisfied. The main direction of IT therefore became the use of high-level programming languages (supporting both general needs and special requirements) and the construction of efficient compilers and interpreters for them.

In 1973, László Kalmár was requested by the Hungarian Academy of Sciences to examine the situation regarding computers and higher-level programming languages (including machine languages) and the progress to be expected in these fields. The goal was for Hungarian IT to find its place and to play an appropriate role in the bright future of IT research and development. It was becoming increasingly more obvious that, like all other countries in Eastern Europe, Hungary had failed to recognize the importance of IT in time, and had not allocated sufficient funds for IT research. This invitation was a sign of appreciation of the leader of one of the few research groups which had been producing results and which had come up with constructive ideas despite the inadequate support.

The development of the various generations of computers up to that time, and the methods of constructing computer architectures, were reviewed in a series of monographs [3]. Naturally, the emphasis was on programming languages and their interpretational possibilities, one of Kalmár's main interests. In the final edition, some 15 years after the planning of the first formula-driven machine he put forward a new proposal for a computer that could be programmed through the use of a high-level machine language.

What were the challenges, to which the new plan was intended to respond?

A look at the algorithmic (problem-oriented) languages reveals that the parts building up the syntax have become clear, and new languages can be designed by combining these parts at will. The block structure and the use of modules were necessities. From the aspect of semantics, the notion of the expression had become very clear, mainly as a result of the increased number of data structures. The programs themselves defined complex data types, and the classical sets of values could no longer be used without appropriate care. The definition and implementation were separated, a situation regarded as normal by today's C++ or Java programmer. It was now obvious that the handling of reference types needed extensive redesigning, a feature applied earlier only for indexed variables and memory assignments. It cannot be claimed that the new plan provided adequate answer to all these questions, but it did so for most of them.

The technology too had been evolving. The problems involving the hardware-manipulated stack had been eliminated. The technique of microprogramming was available, a tool that raised the programming level above that of the architecture. The implementation of a redesigned architecture with high-level machine language again seemed feasible.

In the design stage, it became obvious to Kalmár that a single algorithmic language running on a given architecture could no longer satisfy the users. Compilers were clearly needed. The suggested high-level language was designed to be close to more general programming languages, particularly from the aspect of syntax. This could result in easier compiling, less human work, and lower hardware requirements.

The research group led by Kalmár devised the proposed language. He did not develop a technical plan for the implementation of this language as he had earlier done with the formula-driven machine. He hoped that the various Hungarian workgroups would share their knowledge and that his plans would materialize via such collaboration. In order to verify the language and prove its suitability, he suggested simulation methods. His research group did not lack the necessary knowledge, but the simulator was not built. It is probable that Kalmár's energy was wasted to some extent by his intent on publishing his conceptions in academic and technical circles. The reception was appropriate, and the observations were professional. However, it was at this time that Eastern Europe started planning great developments in the field of IT, and Hungary did not want to be left behind; it therefore adapted to the tendencies determined by "the greats".

What can be said today about formula-driven issues?

Some 30 years ago, IT developed at a very rapid, but quite unexpected tempo. Economizing with hardware resources now belongs to the past; the pace is dictated by the application needs. Many criticize this attitude. Not only has the lowest programming level not risen but it has even become lower, e.g. as a consequence of the RISC technology. The need for the portability of applications over various architectures has nevertheless given rise to a shared-language machine, the Java virtual machine. This has taken place with a different objective and by different means, but its roots are common with those of the formula-driven machine.

References

[1] L. Kalmár, A new principle of construction of logical machines, in Proceedings of 2-e Congres Unternat. de Cybernetique, Namur (1958) 458-463. See in PDF form: http://www.inf.u-szeged.hu/kalmar2005/tcs/kalmar1958.pdf

[2] L. Kalmár, On a digital computer which can be programmed in a mathematical formula language, in Proceedings of 2nd Hungarian Congress of Mathematics, Budapest (1960) Vol. 5, 3-16. See in PDF form: http://www.inf.u-szeged.hu/kalmar2005/tcs/kalmar1960.pdf

[3] Manuscript series under the title "Belső gépi nyelvek", Szeged (1973) (ed.: László Kalmár). Only available in Hungarian.

Szeged, 10 December 2004