

58725

Tomus 3.

Fasciculus 3.



ACTA CYBERNETICA

FORUM CENTRALE PUBLICATIONUM CYBERNETICARUM HUNGARICUM

FUNDAVIT: L. KALMÁR

REDIGIT: F. GÉCSEG

COMMISSIO REDACTORUM

A. ÁDÁM

F. OBÁL

F. CSÁKI

F. PAPP

S. CSIBI

A. PRÉKOPA

B. DÖMÖLKI

J. SZELEZSÁN

T. FREY

J. SZENTÁGOTHAJ

B. KREKÓ

S. SZÉKELY

K. LISSÁK

J. SZÉP

D. MUSZKA

L. VARGA

ZS. NÁRAY

T. VÁMOS

SECRETARIUS COMMISSIONIS

I. BERECSKI

Szeged, 1977

Curat: Universitas Szegediensis de Attila József nominata

ACTA CYBERNETICA

A HAZAI KIBERNETIKAI KUTATÁSOK KÖZPONTI PUBLIKÁCIÓS FÓRUMA

ALAPÍTOTTA: KALMÁR LÁSZLÓ

FŐSZERKESZTŐ: GÉCSEG FERENC

A SZERKESZTŐ BIZOTTSÁG TAGJAI

ÁDÁM ANDRÁS

OBÁL FERENC

CSÁKI FRIGYES

PAPP FERENC

CSIBI SÁNDOR

PRÉKOPA ANDRÁS

DÖMÖLKI BÁLINT

SZELEZSÁN JÁNOS

FREY TAMÁS

SZENTÁGOTHAJ JÁNOS

KREKÓ BÉLA

SZÉKELY SÁNDOR

LISSÁK KÁLMÁN

SZÉP JENŐ

MUSZKA DÁNIEL

VARGA LÁSZLÓ

NÁRAY ZSOLT

VÁMOS TIBOR

A SZERKESZTŐ BIZOTTSÁG TITKÁRA

BERECZKI ILONA

Szeged, 1977. december

A Szegedi József Attila Tudományegyetem gondozásában



FRIGYES CSÁKI
1921—1977

It was an irreplaceable loss to Hungarian science that Professor Frigyes Csáki, a member of the Hungarian Academy of Sciences and an editor of our periodical *Acta Cybernetica* died on August 29, 1977.

He was an outstanding scientist in Control Theory and Computer Science. As an untiring and inspiring teacher he had brought up several generations of electrical engineers.

We honour the memory of Professor Frigyes Csáki.

A note on data base integrity

By A. BENCZÚR and A. KRÁMLI

The interest of the authors in the problem of data base integrity came from a practical task. In 1974 we began to build up a new management information system for the Danube Iron and Steel Works. The data base of this system is placed at the CDC—3300 computer of the Hungarian Academy of Sciences and the system can be accessed through a user terminal UT—200. The processing of the data base is random batch processing with some query capabilities, and the problem of its integrity is very close to that of the online processing.

Based on this experience we are going to give a model for describing and analysing data bases from the point of view of the integrity.

Different approaches to the data base management problems (e.g. Codd's relational data base [1], the network data system recommended by the CODASYL DBTG [2]) first of all give models for describing the structure of data bases, or give proposals for standardizing languages, structures and procedures used in data base management systems (DBMS). A very sophisticated DBMS (based on CODASYL DBTG) is described in the paper of Barbara M. Fossum [3], where one can find a list of routines maintaining and preserving the integrity of the data base. The most general problems of integrity are also described there, and we shall use her terminology like rollback, rerun, recovery, and so on.

In the sequel we shall give a very simplified — but a completely sufficient to treat the problem of integrity — model for physical realization of data bases. Using this model we can investigate the dynamic behaviour of the data base.

Physically the data base is a sequence of elementary data items, (in this paper the elementary data items are the least addressable units on the auxiliary memory devices, i.e., the blocks or pages on the disk).

The physical state S of a data base is given by the sequence $\{P_1, \dots, P_n\}$ of page-addresses and by the sequence $\{C_1, \dots, C_n\}$ of their contents. In our definition the state of a data base, besides the data in traditional sense, includes also the programs handling them.

Definition 1. We shall say that a state $S' = \{\{P'_1, \dots, P'_m\}, \{C'_1, \dots, C'_m\}\}$ is a substate of $S = \{\{P_1, \dots, P_n\}, \{C_1, \dots, C_n\}\}$ ($S' \subseteq S$) iff there exists a subsequence $\{i_1, \dots, i_m\}$ of indices such that $P_{i_j} = P'_j$ and $C_{i_j} = C'_j$ for every $j=1, \dots, m$.

Definition 2. A state \bar{S} is an extension of S iff $S \subseteq \bar{S}$.

The physical state of a data base has to satisfy some rules, which define the syntax, accessing and processing capabilities and so on. We need some further definitions. Let F be a class of "state valued" functions $f(S)$ defined on a set of physical states. The above mentioned rules can be expressed in terms of class F .

Definition 3. We shall say that a state S is consistent with respect to the class F iff $\forall f \in F \ f(S) = S$.

(E.g. if f is a function corresponding to a sort procedure, then S is consistent with respect to f iff it is sorted).

Definition 4. A state S' is called preconsistent with respect to class F , iff there exists a function $g \in F$ such that

- (i) for each $f \in F$, $f(g(S')) = g(S')$,
 (ii) for every g' satisfying (i),
 $g'(S') = g(S)$.

The state $g(S')$ is called the *consistent reorganization* of the preconsistent state S' .

Definition 5. A substate S'' of a preconsistent state S' is a generator, iff it is also a preconsistent state, and they have the same consistent reorganization.

Notice that the inclusion $S'' \subseteq S'$ is not necessary.

The above definitions are suitable not only to describe the static state of a data base, but they give a tool to characterize the dynamic one too. Designing a data base we have to construct the class of functions F in such a way that it would assure the following two capabilities.

Assumption 6. (i) The class F must contain functions which carry out the various update procedures. In terms of our definitions, we consider the state of the data base, before initiating an update run, to be consistent.

(ii) We need to assure with great probability the recovery of the data base in the case of loss or failure of some substate of it. For this reason we have to build up some optimal system of nontrivial generator substates.

Up to this point we have assumed that the preconsistent state becomes consistent in a unique step. But physically it cannot be realized. The real update procedures work in the core memory and the state of the data base changes page by page. So, the realization of a function $g \in F$ is a sequence $S_0, S_1, \dots, S_n = g(S_0)$ of states. (The content of core memory is not included in the notion of a state.)

The defect during an update run, which occurs with a considerable probability, leads usually to the loss of information in the core memory. This raises a special safety problem. In our language, to defend the base against this type of information loss, we have to guarantee the states S_1, \dots, S_{n-1} to be preconsistent. This condition on the state provides the possibility of rerun and in many practical realizations this possibility is combined with rollback facilities on the quick recovery level.

For the evaluation of cost function of different update procedures we make the following assumption: there is a subset of indices i_1, \dots, i_l and a sequence of generators

$$S'_{i_1} \subseteq S_{i_1}, \dots, S'_{i_l} \subseteq S_{i_l}.$$

The substates $S'_{i_1}, \dots, S'_{i_l}$ are the rerun generators and the corresponding functions g_{i_j} determine the rerun procedure.

$$(S'_{i_j} = g_{i_j}(S'_{i_j})).$$

We assume, that the processing cost is a monotonically increasing nonlinear function of the run time, so in order to calculate the average cost we have to determine the probability distribution function of the run-time.

This can be carried out under some conditions on the failure process and the process of changing-points of generators by standard methods of reliability theory.

As an illustration we show how to calculate the probability distribution function of the extra run time caused by random failures occurred during the process S_1, \dots, S_n . We suppose that the following assumptions are fulfilled.

Assumption 7. (i) the failure process $\{\tau_k\}$, where τ_k is the time interval (the length of the time between two consecutive changes is taken for the time unit) between the $(k-1)$ -th and k -th failure, is a sequence of independent identically distributed (i. i. d.) discrete valued random variables with common distribution function F ,

(ii) the first two moments of distribution F are finite,

(iii) the differences $i_k - i_{k-1}$ are equal to a constant d for every $k=1, \dots, l$,

(iv) the rerun procedure $S'_{i_j} \rightarrow g(S'_{i_j}) = S_{i_j}$ does not need extra time.

Let us introduce the random processes $\{\theta_k\}$ and $\{\varepsilon_k\}$ as follows:

$$\theta_1 = \max_{i_1 < \tau_1} i_1,$$

⋮

$$\theta_k = \left(\max_{i_1 < \sum_{j=1}^{k-1} \theta_j + \tau_k} i_1 \right) - \sum_{j=1}^{k-1} \theta_j,$$

$$\varepsilon_k = \sum_{j=1}^{k-1} (\theta_j + \tau_j) - \sum_{j=1}^k \theta_j = \tau_k - \theta_k.$$

The random variable θ_k denotes the time interval between two consecutive regeneration points of the update run, and ε_k the extra run time caused by the k -th failure.

The following two properties are consequences of conditions

(i)–(iv) assumption and the definitions of processes $\{\theta_k\}$, $\{\varepsilon_k\}$:

(v) $\{\theta_k\}$ forms a sequence of i. i. d. random variables having finite first and second moments,

(vi) $\{\varepsilon_k\}$ forms a sequence of i. i. d. random variables bounded by d .

We have to determine the probability distribution function of the random variable $\eta_n = \sum_{k=1}^{v_n-1} \varepsilon_k$, where v_n is the first moment m for which $\sum_{k=1}^m \theta_k \geq n$. The asymptotic behaviour of the distribution function of random variables η_n , when $n \rightarrow \infty$, can be expressed by the following theorem:

Theorem 1. If $F(d) \neq 1$, then $E(\theta_1) \neq 0$ and

$$\lim_{n \rightarrow \infty} P \left(\frac{\eta_n - \frac{n}{E(\theta_1)} E(\varepsilon_1)}{\sqrt{\frac{n}{E(\theta_1)} E(\varepsilon_1 - E(\varepsilon_1))^2}} < x \right) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-\frac{x^2}{2}}.$$

Proof. It follows from (v) and the weak law of large numbers that for $n \rightarrow \infty$, $\frac{\eta_n}{n} \rightarrow \frac{1}{E(\theta_1)}$ in probability. Thus taking into account property (vi) we can apply to $\{\eta_n\}$ Anscomb's central limit theorem for the sum of a random number of random variables (see Rényi [4]).

The difference between the long and quick recovery problems can be summarized as follows: in the quick recovery we preserve generator substates of preconsistent states during an update run, while for solving the long recovery problem we have to preserve (generally duplicate) generator substates of a consistent state before a sequence of update runs. Between two duplications we collect the changes in such a way, that the preserved generators of the original state and the changes together compose a generator of the present state.

So the reliability of the system is the product of the reliability of the preserved generator and that of the preserved changes. The cost of this recovery system consists of the cost of the periodical duplications, the cost of collecting the changes (proportional to the update time) and the average cost of the recovery. The probabilistic treatment of this process is analogous to that of the quick recovery.

Finally, we mention some special problems arising in the use of the operating system MASTER of computer CDC-3300. (See [5]). We must handle the SCHRATCH-pool as an extension of core memory and not as a part of the physical state. The only exception is the INP-file for not DIRECT input JOB-s. In this case the operating system ensures the restart of the JOB using the INP-file as a part of a generator, when an AUTOLOAD has occurred during the run of the JOB.

This facility was exploited only for automatic restart of update processes by preserving only the control cards. The input data were put into the data base before initiating the update system in strict sense.

This method can serve to collect the changes for long time recovery and in some cases to obtain a part of a generator for quick recovery.

The update system of the realized DBMS mentioned at the beginning of our paper consists of 40 processes corresponding to the various types of data. Each process is organized in such a way that their repetition, or rollback, or continuation when it is interrupted would be available. The organization of the correct runs of these processes is solved by a special program (TASK), which calls the update tasks in correct sequence, determines the input data, controls their run and, in the case of a restart calls the necessary rollback procedures and continues the run from the interrupted task.

The long recovery is based on the File Back Up system of MASTER.

Abstract

The integrity and reliability problems of data base systems arise in computer praxis and theoretical investigations too. The authors give a statistical model for the description of data bases, which is sufficient to treat the integrity problem.

COMPUTER AND AUTOMATION INSTITUTE
HUNGARIAN ACADEMY OF SCIENCES
H-1502 BUDAPEST, HUNGARY

References

- [1] CODD, E. F., A relational model of data for large shared data bases, *Comm. ACM*, v. 13, No. 6, 1970.
- [2] *CODASYL committee data base task group report*, Association for Computing Machinery, April 1971.
- [3] FOSSUM, B. M., Data base integrity as provided for by a particular data base management system, *Proceedings of IFIP Working Conference on Data Base Management*, Cargese, Corsica France, 1—5 April 1974.
- [4] RÉNYI, A., On the central limit theorem for the sum of a random number of independent random variables, *Acta Math. Acad. Sci. Hungar.*, v. 11, 1960, pp. 97—102.
- [5] CDC 3170—3300—3500 computer systems MASTER version, 4.0. *Reference Manual CDC*, 60415100, No. 1973.

(Received March 11, 1976)



On some open problems of applied automaton theory and graph theory (suggested by the mathematical modelling of certain neuronal networks)

By A. ÁDÁM

Introduction

The branch of investigations to which this paper is devoted was initiated by U. Kling and Gy. Székely [14]. They studied some kind of electrical networks simulating certain nervous activities, and facilitating the quantitative treatment of such phenomena. The description of structure and function of such networks was continued in the articles [2], [7], [3] etc., using mathematical tools.

This paper contains a (more or less detailed) survey of the mathematical considerations mentioned above and — primarily — a list of numerous open problems. The article consists of four chapters. In Chapter I certain finite directed graphs are considered. The questions raised here may mostly be viewed as “variations on the theme” of describing graph classes each of which is a natural extension of the class of single cycles (in one or another sense). Chapter II starts with a systematization of the behaviour of autonomous continuous automata and, as a par-

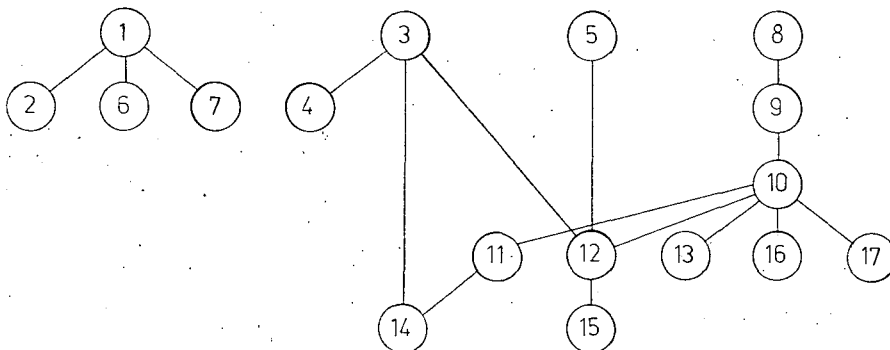


Fig. 1
Subordination of the sections of the paper

ticular case, deals chiefly with the network notions without supposing any special graph structure. In Chapter III the behaviour of networks with special structure is treated and the problem of speed of propagating actions is posed. In Chapter IV some questions of stochastic behaviour of networks are touched.

The first half of the paper contains a number of assertions, too. Some of them (e.g. Proposition 8) are easy consequences of the concepts defined or are "folkloristically" known to the specialists of the topics (as Propositions 12—14). Only the statements of § 4 and § 7 could be regarded as (more or less) original results.

The subject of Chapter I has its own importance in the theory of graphs. The considerations referred to in Chapters II—IV may be estimated rather as an attempt how a certain type of questions admits an exact mathematical treatment, than settled, definitive scientific advances.

The exposed problems are partly strictly determined ones (as Problem 14), partly proposals for making researches in some intuitively encircled field (e.g. Problem 7), or of transitional character between these extremities.

In the assembling of the material of this paper I was not free from some subjectiveness. The variety of ideas in the first chapter has followed from my affection for structural descriptions; on the other hand, the fact that I am no probability theorist has implied that the (very important) questions of stochastic behaviour appear in a smaller extent than they would deserve.

I. Structural problems (Problems concerning graphs)¹

§ 1.

In this § we deal always with strongly connected directed finite graphs (see Chapter 16 of [12]). The graph with one vertex and without edges is excluded. We do not allow loops and parallel edges with the same orientation. By a *cycle* (of a graph) we mean a circuit (without repeated vertices) along which each edge e is passed through in sense of the orientation of e . For any edge e the number of cycles containing e is denoted by $Z(e)$; similarly, $Z(A)$ is the number of cycles in which a vertex A occurs. The strong connectedness implies $Z(e) \geq 1$ for every edge and $Z(A) \geq 1$ for every vertex of the graph. (Conversely, if a connected graph is not strongly connected, then $Z(e) = 0$ for some edge of it.)

If a graph G can be represented as the union of two subgraphs G_1, G_2 (each having at least one edge) such that G_1 and G_2 have only one vertex A in common, then A is called a *cut vertex*². If a vertex A of a graph G is contained in every cycle of G , then A is called a *pancyclic vertex*.

Now let four properties (α), (β), (γ), (δ) of graphs be defined. (For the sake of brevity we shall say e.g. " $(\alpha\beta)$ -graph" instead of "graph satisfying (α) and (β)".)

- (α) $Z(e) \leq 2$ for every edge e of the graph,
- (β) $Z(A) \leq 2$ for every vertex A of the graph,
- (γ) the graph has no cut vertex,

¹ Chapter I has already been propagated in preprint form under the title "Some open questions concerning finite directed graphs".

² The term "articulation vertex" is also used.

(δ) the graph has a pancyclic vertex.

The main question to be proposed in this § is:

PROBLEM 1 (see [3]). Describe the structure of all (α)-graphs.

In some particular cases, Problem 1 has been solved. § 3 of [3] deals with the structural description of (β)-graphs, and any (β)-graph is clearly an (α)-graph. On the other hand, [4] is devoted to the description of ($\alpha\gamma\delta$)-graphs and the extension of this to ($\alpha\delta$)-graphs.

It seems that the difficulty of solving Problem 1 lies in getting an overview of the ($\alpha\gamma$)-graphs, therefore we formulate separately

PROBLEM 2. Describe the structure of all ($\alpha\gamma$)-graphs.

In § 7 we shall give a relative solution of Problem 1 presupposing that Problem 2 is settled.

§ 2.

The terminology of § 1 is continued, especially, the condition of strong connectedness is maintained. Let us recall Property (δ) and define two related properties:

(ϵ) The graph has a vertex A such that

for any choice of the vertex B there is a cycle containing both A and B , and for any choice of the edge e there is a cycle containing both A and e .

(ζ) Each pair of cycles has at least one vertex in common.

It is trivial that each (δ)-graph is an ($\epsilon\zeta$)-graph and each ($\epsilon\zeta$)-graph is an (ϵ)-graph. The examples on Fig. 2 show that these inclusions (concerning the classes of (δ)-graphs, ($\epsilon\zeta$)-graphs and (ϵ)-graphs) are proper³. The connection of (ϵ) and (ζ) is questionable as follows:

PROBLEM 3. Does there exist a (ζ)-graph which does not satisfy (ϵ)?

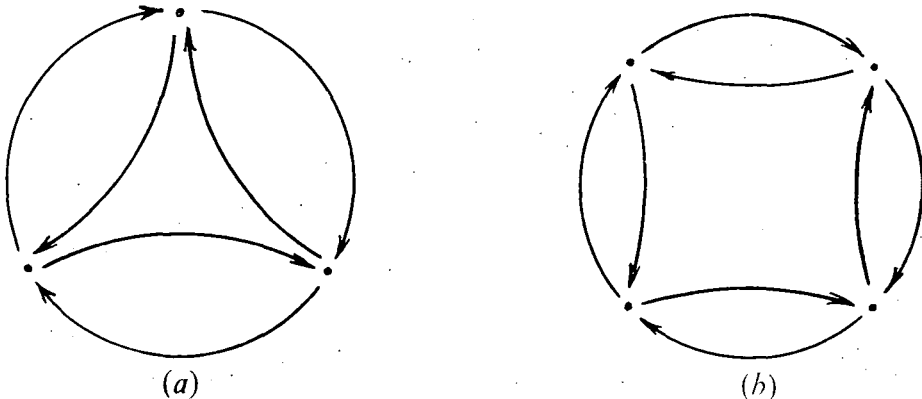


Fig. 2

³ The graph on Fig. 2/b was called to my attention by Dr. B. Zelinka.

§ 3.

Let H be a set with $n (\geq 2)$ elements. A permutation α of H is called *cyclic* if α fixes q elements and permutes the remaining $n-q$ ones cyclically (where $0 \leq q \leq n-2$ or $q=n$). α is *fully cyclic* if it is cyclic with $q=0$. By a *fully cyclic automorphism* of a graph such an automorphism is understood which acts as a fully cyclic permutation on the set of vertices.

In contrasting with § 1, now we do not restrict ourselves to connected graphs⁴. Let us choose some integers $n, k, m_1, m_2, \dots, m_k$ such that

$$(3.1) \quad k < n, \quad 1 \leq m_1 < m_2 < \dots < m_k < n.$$

We define the (labelled) graph $G = G(n; m_1, m_2, \dots, m_k)$ as follows:

the vertices of G are denoted by P_1, P_2, \dots, P_n ;
the edge from P_i to P_j (where $1 \leq i \leq n, 1 \leq j \leq n$) exists in G if and only if there is a number h ($1 \leq h \leq k$) for which the congruence

$$i - j \equiv m_h \pmod{n}$$

holds.

Proposition 1. *A graph G can be expressed as $G(n; m_1, m_2, \dots, m_k)$ if and only if G has a fully cyclic automorphism.*

Proof. Let $G(n; m_1, m_2, \dots, m_k)$ be considered. The following (fully cyclic) vertex permutation α is obviously an automorphism:

$$\alpha(P_i) = \begin{cases} P_{i+1} & \text{if } 1 \leq i < n, \\ P_1 & \text{if } i = n. \end{cases}$$

Conversely, suppose that a graph G (having n vertices) possesses a fully cyclic automorphism α . For an arbitrary vertex A , let us introduce the notation

$$A = P_1, \quad \alpha(A) = P_2, \quad \alpha^2(A) = P_3, \quad \alpha^3(A) = P_4, \quad \alpha^{n-1}(A) = P_n$$

and let m_1, m_2, \dots, m_k be defined by the conditions

$$m_1 < m_2 < \dots < m_k,$$

the edges $\overline{P_n P_{m_1}}, \overline{P_n P_{m_2}}, \dots, \overline{P_n P_{m_k}}$ exist in G , and

there are no other edges from P_n .

It is easy to see that G equals $G(n; m_1, m_2, \dots, m_k)$. \square

For given n and k , two sequences (m_1, m_2, \dots, m_k) and $(m'_1, m'_2, \dots, m'_k)$ (fulfilling (3.1)) are called *equivalent* (for n) if there exists a number r ($1 \leq r < n$) and a permutation π of the set $\{1, 2, \dots, k\}$ such that r is relatively prime to n and the congruences

$$(3.2) \quad r m_1 \equiv m'_{\pi(1)}, \quad r m_2 \equiv m'_{\pi(2)}, \quad \dots, \quad r m_k \equiv m'_{\pi(k)}$$

are valid modulo n .

⁴ The meaning of "graph" is unchanged in any other respect.

Proposition 2. *The equivalence defined above is a reflexive, symmetric and transitive relation (in the set of all sequences (m_1, m_2, \dots, m_k) when n and k are fixed).*

In the proof we shall use the fact that the residue classes, consisting of numbers relatively prime to n , form a multiplicative group.

The reflexivity holds since 1 may be chosen for r . — If some r, π establish a connection of type (3.2), then the solution r' ($1 \leq r' < n$) of $rr' \equiv 1 \pmod{n}$ and π^{-1} establish a connection between the two sequences having interchanged roles. — If $rm_h \equiv r'_{\pi(h)}$ and $r'm'_h \equiv m''_{\pi'(h)} \pmod{n}$, then let r'' ($1 \leq r'' < n$) be defined by $r'' \equiv rr' \pmod{n}$; it follows $r''m_h \equiv r'rm_h \equiv r'm'_{\pi(h)} \equiv m''_{\pi'(h)} \pmod{n}$ (where h may be $1, 2, \dots, k$). \square

Next we state an evident assertion:

Proposition 3. *If two graphs are described in terms of the formalism $G(n; m_1, m_2, \dots, m_k)$ and they are isomorphic, then n and k are common.* \square

Proposition 4. *If (with the same n and k) the sequences (m_1, m_2, \dots, m_k) and $(m'_1, m'_2, \dots, m'_k)$ are equivalent, then the graphs $G = G(n; m_1, m_2, \dots, m_k)$ and $G' = G(n; m'_1, m'_2, \dots, m'_k)$ are isomorphic.*

Proof. To an arbitrary vertex P_i of G , let $\beta(P_i)$ be the vertex $P'_{i'}$ of G' whose subscript is defined by $ri \equiv i' \pmod{n}$ where the equivalence is established by r . We may check that β is an isomorphism. \square

Since Propositions 3 and 4 do not determine fully when two graphs in question are isomorphic, we raise

PROBLEM 4. *Let a condition for two sequences $(m_1, m_2, \dots, m_k), (m'_1, m'_2, \dots, m'_k)$ be stated which is necessary and sufficient in order $G(n; m_1, m_2, \dots, m_k)$ and $G(n; m'_1, m'_2, \dots, m'_k)$ be isomorphic.*

I have conjectured [1] that the converse of Proposition 4 is also valid, i.e. that the equivalence is necessary for isomorphism, too*. The counter-examples due to Elspas and Turner [10] show that the conjecture is false in the class of all graphs having a fully cyclic automorphism; namely, $G(8; 1, 2, 5)$ is isomorphic to $G(8; 1, 5, 6)$ and $G(16; 1, 2, 7, 9, 14, 15)$ is isomorphic to $G(16; 2, 3, 5, 11, 13, 14)$ although neither $(1, 2, 5)$ and $(1, 5, 6)$ (for $n=8$) nor $(1, 2, 7, 9, 14, 15)$ and $(2, 3, 5, 11, 13, 14)$ (for $n=16$) are equivalent.

It was proved by Đoković, Elspas, Toida and Turner (see [9], [10], [16]) that my conjecture is valid within each of the following four subclasses of the mentioned class:

- (i) the class of graphs the number of whose vertices is a prime,
- (ii) the class of graphs whose adjacency matrices have non-repeated eigenvalues only,
- (iii) the class of graphs with $k=3, m_1+m_3=n$ and $m_2=n/2$,
- (iv) the class of graphs with $k=4, m_1+m_4=m_2+m_3=n$ and $(m_1, n)=(m_2, n)=1$ (the parentheses denote here largest common divisor).

They use — somewhat surprisingly — mostly tools lying outside graph theory (e.g. techniques of matrix theory).

* Remark added in proof (November 21, 1977). For category-theoretical generalizations see [18].

§ 4.

This § is⁵ a continuation of the preceding one. Our present aim is to re-formulate Problem 4 in terms of automorphisms. For the sake of simplicity, we do not make a distinction between a permutation π of the set $\{1, 2, \dots, n\}$ and the permutation of vertices defined by $P_i \rightarrow P_{\pi(i)}$.

Let a vertex permutation π of the graph $G(n; m_1, m_2, \dots, m_k)$ be called *special* when $\pi\alpha\pi^{-1}$ is an automorphism where

$$\alpha(i) = \begin{cases} i+1 & \text{if } 1 \leq i < n \\ 1 & \text{if } i = n \end{cases}$$

Proposition 5. Consider a vertex permutation π of $G = G(n; m_1, m_2, \dots, m_k)$. Let us introduce another labelling P'_1, P'_2, \dots, P'_n of the vertices of G by the equalities $P'_i = P_{\pi(i)}$. If π is special, then G (provided with the new notation) equals

$$G(n; \pi^{-1}(m_1 + \pi(n)), \pi^{-1}(m_2 + \pi(n)), \dots, \pi^{-1}(m_k + \pi(n)))$$

where the k sums (after the semicolon) are thought to be reduced modulo n and ordered increasingly. If π is not special, then the new notation of vertices does not allow to write G as $G(n; m_1, m_2, \dots, m_k)$ (for any choice of the sequence (m_1, m_2, \dots, m_k)).

Proof. Let π be special. This means that $\pi\alpha\pi^{-1}$ is an automorphism. The deductions $\pi(P_i) = P_{\pi(i)} = P'_i$ and

$$\pi\alpha\pi^{-1}(P'_i) = \pi\alpha(P_i) = \pi(P_{i+1}) = P'_{i+1}$$

show⁶ that $\pi\alpha\pi^{-1}$ acts in the same manner as α in the sufficiency proof of Proposition 1. There are evidently k edges incoming to $P'_n (= P_{\pi(n)})$, these edges are outgoing from the vertices

$$P_{\pi(n)+m_1} = P'_{\pi^{-1}(\pi(n)+m_1)}, \quad P_{\pi(n)+m_2} = P'_{\pi^{-1}(\pi(n)+m_2)}, \quad \dots, \quad P_{\pi(n)+m_k} = P'_{\pi^{-1}(\pi(n)+m_k)}$$

where the sums are meant mod n .

Conversely, suppose that π is not special. There is a pair (P_i, P_j) such that exactly one of the edges $\overrightarrow{P_i P_j}$ and $\overleftarrow{(\pi\alpha\pi^{-1}(P_i))(\pi\alpha\pi^{-1}(P_j))}$ exists. We have the equalities

$$\begin{aligned} P_i &= P'_{\pi^{-1}(i)}, \quad P_j = P'_{\pi^{-1}(j)}, \\ \pi\alpha\pi^{-1}(P_i) &= \pi\alpha(P_{\pi^{-1}(i)}) = \pi(P_{1+\pi^{-1}(i)}) = P'_{1+\pi^{-1}(i)}, \\ \pi\alpha\pi^{-1}(P_j) &= P'_{1+\pi^{-1}(j)} \end{aligned}$$

(the subscripts have sometimes to be reduced mod n), thus the fully cyclic permutation $P'_i \rightarrow P'_{i+1}$ is not an automorphism. \square

Proposition 6. The graphs $G = G(n; m_1, m_2, \dots, m_k)$ and $G' = G(n; m'_1, m'_2, \dots, m'_k)$ are isomorphic if and only if there exists a pair (π, ϱ) satisfying the following three properties:

⁵ The reader may neglect this § unless he is particularly interested in Problem 4.

⁶ We omit the separate treatment of the case $i=n$.

π is a permutation of the set $\{1, 2, \dots, n\}$ and — as a permutation of the vertex set of G — π is special,
 ϱ is a permutation of the set $\{1, 2, \dots, k\}$,
 the congruence

$$\pi(m'_{\varrho(h)}) \equiv m_h + \pi(n) \pmod{n}$$

holds for each h ($1 \leq h \leq k$).

Proof. Let β be an isomorphism of G onto G' , denote by γ the mapping of the vertex set of G onto the vertex set of G' satisfying $\gamma(P_i) = P'_i$ ($1 \leq i \leq n$). We introduce new notations $P''_1, P''_2, \dots, P''_n$ by the formula $\beta(P''_i) = P'_i$, we can now write also G in the form $G(n; m'_1, m'_2, \dots, m'_k)$. Proposition 5 is applicable (with $\beta^{-1}\gamma$ in the role of π).

Assume the existence of π and ϱ that satisfy the conditions. Proposition 5 assures that G can be made isomorphic to $G(n; m'_1, m'_2, \dots, m'_k)$ by introducing the notations $P''_i = P_{\pi(i)}$ ($1 \leq i \leq n$). \square

PROBLEM 5. Let a method be given which, for an arbitrary graph $G = G(n; m_1, m_2, \dots, m_k)$ gives a survey of all (different) systems

$$\{\pi^{-1}(m_1 + \pi(n)), \pi^{-1}(m_2 + \pi(n)), \dots, \pi^{-1}(m_k + \pi(n))\}$$

where π runs through the special permutations of the vertex set of G and the sums are meant modulo n . \square

By virtue of Proposition 6, a solution of Problem 5 would imply the solution of Problem 4.

§ 5.

We study finite directed graphs without loops and pairs of parallel edges (with coinciding or opposite orientation). We denote by $M(G)$ the length of a shortest cycle of the graph G . If the number k satisfies⁷ $2 \leq k < M(G)$, then we assign a new graph $\mathfrak{A}_k(G)$ to the graph G in the following way:

the vertex set of $\mathfrak{A}_k(G)$ equals the vertex set of G ,

the edge \overline{AB} exists in $\mathfrak{A}_k(G)$ if and only if ($A \neq B$ and) there is a path in G from A to B the length of which is smaller than k .

$\mathfrak{A}_2(G)$ is G itself. If G is a cycle of length n , then $\mathfrak{A}_k(G) = G(n; 1, 2, \dots, k-1)$ where the right-hand side is to be understood as in § 3.

Proposition 7. If G is a subgraph of H (with the same vertex set as H), then there exists at most one number k fulfilling $\mathfrak{A}_k(G) = H$.

Proof. Let H equal $\mathfrak{A}_k(G)$ for some k . Denote by $A_1, A_2, \dots, A_{M(G)}$ the vertices of a shortest cycle of G (in the natural ordering). Suppose $2 \leq k' < M(G)$ and $k' \neq k$, denote by k'' the larger of k and k' . The edge $\overrightarrow{A_{M(G)}A_{k''}}$ exists in precisely one of $\mathfrak{A}_k(G)$ and $\mathfrak{A}_{k'}(G)$, thus $\mathfrak{A}_k(G) \neq \mathfrak{A}_{k'}(G)$. \square

⁷ The letter k is now used in another sense, than in the previous sections.

Let C be a class consisting of directed graphs. Then we denote by $\mathfrak{U}(C)$ the class of all graphs $\mathfrak{U}_k(G)$ where G runs through the members of C and, for any G , k runs through the numbers satisfying $2 \leq k < M(G)$.

By virtue of this definition, every member H of $\mathfrak{U}(C)$ has at least one subgraph $G(\in C)$ such that $\mathfrak{U}_k(G)=H$ for some k . C is called a *decomposition class* if every member H of $\mathfrak{U}(C)$ can be represented with exactly one $G(\in C)$ in the form $\mathfrak{U}_k(G)$.

Proposition 8. *The class C consisting of all cycles is a decomposition class.*

Proof. Let A be a vertex of a member H of $\mathfrak{U}(C)$. Denote the outdegree of A by $\varrho(A)$ and the set of end vertices of the edges outgoing from A by $\sigma(A)$. Then $\varrho(A)$ is common for the vertices of H and — denoting it by ϱ — $|\sigma(A)| = \varrho$.

Let G be an arbitrary cycle such that $\mathfrak{U}_k(G)=H$. It may be seen easily that $\varrho = k - 1$, furthermore,

$$|\sigma(A) \cap \sigma(B)| = k - 2$$

if A and B are adjacent vertices in G but⁸

$$|\sigma(A) \cap \sigma(B)| \leq k - 3$$

for any other choice of A and B ($A \neq B$).

We have reconstructed the pair (G, k) in terms of H only. \square

Let the notion of (β) -graphs be recalled (see § 1).

PROBLEM 6. *Prove or disprove that the family of (β) -graphs is a decomposition class.*

This problem was raised in § 4 of [3] as Conjecture 3 together with some related conjectures.

Problem 6 is a particular case of the subsequent question (of rather heuristic than exact nature):

PROBLEM 7. *Let us determine decomposition classes, comprehensive as far as possible, among the finite directed graphs.*

§ 6.

A graph G is called *cyclically simple* if the intersection of any pair of different cycles of G is (empty or) a path. The graph on Fig. 3 is not cyclically simple because the intersection of the cycles $(ABDEF)$ and

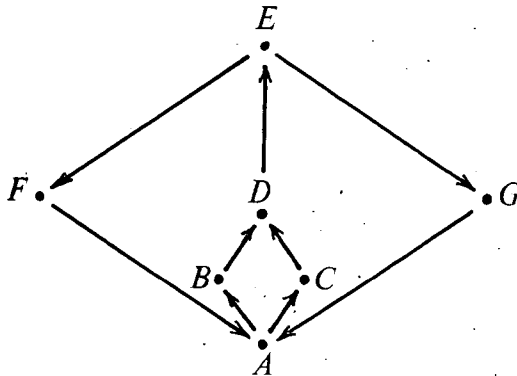


Fig. 3

⁸ We utilize here the fact $k < n$ where n is the length of G .

(ACDEG) consists of two paths (of lengths zero and one, resp.) being not connected with each other.

The future development of the methods of analyzing directed graphs structurally, with a particular emphasis to their cycles (see [3], [4], [5]), will perhaps enable us to make remarkable attacks towards the following general research direction:

PROBLEM 8. Let the structure of all cyclically simple graphs be described.

I note that [5] terminates with an open question. This problem is somewhat particular, this fact and the lengthiness of the previous definitions (before all, the B-constructibility) do not permit to recapitulate it here within reasonable size.

§ 7.⁹

Let V be a finite set and

$$\mathfrak{S} = \{H_1, H_2, \dots, H_t\} \quad (t \geq 1)$$

be a family of subsets of V . The pair (V, \mathfrak{S}) is called a *hyper-tree* if the following four conditions are fulfilled:

- (A) $H_1 \cup H_2 \cup \dots \cup H_t = V$,
- (B) $|H_i| \geq 2$ for each i ($1 \leq i \leq t$),
- (C) $|H_i \cap H_j| \leq 1$ whenever $1 \leq i < j \leq t$,
- (D) to each pair (i, j) (where $1 \leq i < j \leq t$) there exists precisely one sequence

$$i = i_0, i_1, i_2, \dots, i_m = j \quad (m \geq 1)$$

satisfying the properties (i), (ii):

- (i) i_0, i_1, \dots, i_m are pairwise different numbers chosen from the set $\{1, 2, \dots, t\}$,
- (ii) $H_{i_{p-1}} \cap H_{i_p} \neq \emptyset$ whenever $1 \leq p \leq m$.

Let some evident consequences of the above definition of hyper-tree be stated. $H_i \cap H_j = \emptyset$ if $i \neq j$ (by (B) and (C)). The intersection in (ii) has exactly one element (by (C)). If $2 \leq p+1 < q \leq m$, then $H_{i_p} \cap H_{i_q}$ is empty in (D) (by the unicity of the sequence).

The elements of V are called the *vertices* of the hyper-tree, the members of \mathfrak{S} are the *hyper-edges* of it.

CONSTRUCTION. The construction consists of three steps.

Step 1. Let a hyper-tree (V, \mathfrak{S}) be considered. We assign to any vertex $A (\in V)$ an $(\alpha\gamma)$ -graph $v(A)$. At the beginning of the procedure, $v(A)$ and $v(B)$ are viewed to be disjoint if A, B are different vertices.

Step 2. Let $\mu(A, H_i)$ be a mapping such that

- (a) μ is defined on the set of pairs $A (\in V), H_i (\in \mathfrak{S})$ such that $A \in H_i$,
- (b) the value of $\mu(A, H_i)$ is a vertex of $v(A)$, and
- (c) $i \neq j$ implies $\mu(A, H_i) \neq \mu(A, H_j)$ (where, of course, $A \in H_i \cap H_j$).

⁹ This § is addressed only to readers interested in Problem 1 or how graphs are built up from their blocks.

Step 3. For any $H_i (\in \mathfrak{H})$, let us identify all the vertices $\mu(A, H_i)$ with each other (A runs through the elements of H_i). Denote the resulting graph by G .

The description of the Construction is completed.

In Step 3, any $v(A)$ is embedded into G . We denote by $v^*(A)$ the result of this embedding. The graphs $v^*(A)$ are not necessarily disjoint, unlike the $v(A)$'s.

Proposition 9. *Let P be a vertex of the graph G (constructed above). P is a cut vertex of G if and only if there is a $H_i (\in \mathfrak{H})$ such that P is the result of the identification of the vertices $\mu(A, H_i)$ (in sense of Step 3).*

Proof. We use the following characteristic property of cut vertices: P is a cut vertex if and only if there exist two vertices Q, R such that Q and R are adjacent to P and every chain between Q and R contains P .

Necessity. Suppose that P has not been produced by identification. Then P is a vertex of some (well-determined) $v^*(A)$ and any vertex adjacent to P (in G) is in the same $v^*(A)$. $v^*(A)$ is an $(\alpha\gamma)$ -graph, hence it has no cut vertex. For every choice of Q and R , these vertices may be connected by a chain within $v^*(A)$ which does not pass through P ; the same holds obviously in G , too.

Sufficiency. Assume that P originates from identification. Let us choose two elements A and B of H_i . There is a vertex Q in $v^*(A)$ and a vertex R in $v^*(B)$ such that they are adjacent to P , i.e. they can be connected by a chain a_1 whose vertices are Q, P, R (in this ordering). Suppose that there exists a chain a_2 (in G) which connects Q and R and does not contain P . The union of a_1 and a_2 is a circuit a . Passing along a and considering the common vertices of the subgraphs of type $v^*(A)$, we can form a sequence

$$H_{i_1}, H_{i_2}, \dots, H_{i_w} \quad (w \geq 2)$$

of some members of \mathfrak{H} such that every pair of neighbouring members of the sequence is a pair of distinct and non-disjoint elements, moreover also H_{i_1} and H_{i_w} have an element in common. If $w > 2$, then this contradicts the condition (D) in the definition of hyper-trees. The case $w=2$ is impossible by (C). \square

Proposition 9 implies immediately

Proposition 10. *The blocks of G coincide with the subgraphs $v^*(A)$ where A runs through the elements of V . \square*

Proposition 11. *Every graph produced by our Construction is an (α) -graph and all (α) -graphs can be represented in this manner.*

Proof. Any circuit (and, particularly, any cycle) of G is entirely included in some $v^*(A)$, thus the Construction leads to an (α) -graph if all the $v(A)$'s were (α) -graphs.

Conversely, consider the blocks of an (α) -graph G . Every block is an $(\alpha\gamma)$ -graph. If

we assign an element A to any block,

V denotes the set of the A 's ($|V|$ is the number of blocks),

we assign a H_i to the situation (whenever it occurs) that the same vertex of G occurs in two or more blocks, and

\S denotes the family of H_i 's ($|\S|$ is the number of cut vertices of G), then the conditions of the Construction are obviously satisfied. \square

Remark. In this \S we have utilized only very few properties of the (α) -graphs. Let us consider an arbitrary class of connected graphs (directed or non-directed) — say, the (τ) -graphs — such that a graph G is a (τ) -graph precisely if each block of G is again a (τ) -graph.¹⁰ Then the connection of (τ) -graphs and $(\tau\gamma)$ -graphs can be described analogously to the previous treatment.

II. Problems on the functioning of networks

§ 8.

By an *autonomous continuous automaton* we understand in the sequel a partial function $\varphi(x, t)$ fulfilling the seven conditions as follows:

- (i) x runs through the elements of a set X ,
- (ii) t runs through the non-negative real numbers,
- (iii) each value of φ is an element of X ,
- (iv) for any fixed $x(\in X)$, either $\varphi(x, t)$ is defined for every t or there exists a bound $b_x(\geq 0)$ (depending on x) such that $\varphi(x, t)$ is defined precisely when $0 \leq t < b_x$,
- (v) $\varphi(x, 0) = x$ if the left-hand side is defined (i.e. if $b_x > 0$),
- (vi) whenever $\varphi(x, 0)$ is defined, then there exists an $r_x(> 0)$ (depending on x) such that either
 - $0 < t \leq r_x$ implies $\varphi(x, t) = x$ or
 - $0 < t \leq r_x$ implies $\varphi(x, t) \neq x$,
- (vii) the equality

$$\varphi(x, t_1 + t_2) = \varphi(\varphi(x, t_1), t_2) \quad (8.1)$$

is required for every triple $x(\in X)$, $t_1(\geq 0)$, $t_2(\geq 0)$ (in such a sense that either both sides of (8.1) are defined or none of them).

Now we turn to a heuristical explanation of what the definition of autonomous continuous automata expresses. The second variable t of φ is interpreted as time. The first variable x corresponds to the possible states of a system (working in time). The function φ itself has the following meaning: whenever the system is in a state x at the initial instant 0 and the function value $\varphi(x, t)$ is defined, then the system will take the state $\varphi(x, t)$ at the instant t . (The case when $\varphi(x, t)$ is undefined means that our mathematical model is unable to say what will be the state of the system at t .)

In (iv) three possibilities are allowed, namely, either b_x is 0 or b_x is a positive real number or b_x does not occur. The situation $b_x = 0$ means that, at least in sense of the mathematical treatment of the system's behaviour, the system is not capable to take the state x . (In the following §§ this will arise when the edge \overline{PQ} exists in a graph, and the state x attributes the value 1 to P and a value lying in the interval¹¹ $(0, 1]$ to Q .) If a positive b_x exists, then our mathematical apparatus

¹⁰ Of course, the blocks are $(\tau\gamma)$ -graphs.

¹¹ By $(0, 1]$ the set of instants t fulfilling $0 < t \leq 1$ is denoted.

describes the functioning of the system not longer than within a time interval having a finite (positive) length. When b_x is not defined, then our model is able to prognosticate the system's states for every future instant.

The aim of the condition (vi) is to exclude the automata whose functioning is very irregular. It does not permit the occurrence of the following (abnormal enough) situation: for any positive r (however small it be!) there exist two instants t, t' in the interval $(0, r]$ such that the state of the system at t is x and its state at t' differs from x (i.e., roughly speaking, the state x is densely mixed with other state(s) in the neighbourhood of the initial instant). — Such situations can scarcely arise in the work of real systems, but they are logically imaginable.

Condition (vi) is rather of technical character; in contrast with this, (vii) expresses an important characteristic feature of the considered automata. The notation $\varphi(x, t)$ (containing no symbol denoting effects which come from outside!) is interpreted that the automaton works autonomously; in addition to this, (vii) postulates that the laws of its functioning do not change with time, in other words, the distinguished role of the initial instant is abolished and t may be viewed as the length of a time interval (situated anywhere in the non-negative semiaxis). (We can also say, on the basis of (vii), that the system neither oldens nor learns.)

In accordance with the above intuitive considerations, we introduce the following terminologies. The elements of X are also called *states* (of the automaton A , to be defined later). The variable t is interpreted as time, thus its values are called *instants*. If $\varphi(x, 0)$ is defined, then x is called a *permitted state*. The condition (vi) is said the *quasi-continuity* of φ , (vii) is said the *homogeneity* of φ in time.

A pair $A = (\varphi, x_0)$ is called an *initial autonomous continuous automaton* (or, for the sake of brevity, an *initial automaton*) if φ is an autonomous continuous automaton and x_0 is a (fixed) permitted element of X . We call x_0 the *initial state* of A . $\varphi(x_0, t)$ is said the state of A at the instant t .

If x is a state of φ , then we denote by H_x the set of positive numbers t_0 satisfying $\varphi(x, t_0) = x$. — We write also shortly H (instead of H_x) when only one state is considered. If some states x_1, x_2, \dots are viewed, then the simple notations H_1, H_2, \dots may be used (instead of H_{x_1}, H_{x_2}, \dots , resp). Similar simplifications will be used for other quantities depending on states too.

Proposition 12. *If x is a state of the autonomous continuous automaton φ such that H_x is neither empty nor the set of all positive numbers, then H_x contains a smallest element p_x .*

Proof. Suppose that H is not empty and (the existence of smallest element does not hold, i.e.) for every $t_0 (\in H)$ there is a t_1 satisfying both $0 < t_1 < t_0$ and $t_1 \in H$. Our aim is to show that every positive number belongs to H . By iterating the step of determining t_1 , we get an infinite decreasing sequence

$$t_0 > t_1 > t_2 > t_3 > \dots$$

consisting of elements of H . The difference $d_i = t_i - t_{i+1}$ converges to 0 if i tends to the infinity. We have

$$\varphi(x, d_i) = \varphi(\varphi(x, t_{i+1}), d_i) = \varphi(x, t_{i+1} + d_i) = \varphi(x, t_i) = x$$

(since φ is homogeneous in time) for each i , thus there exists an r such that $\varphi(x, t) = x$ whenever $0 < t \leq r$ (by the quasi-continuity of φ).

Let now t^* be an arbitrary positive number. There is an integer u and a (real) number v such that $t^* = ur + v$ and $0 \leq v < r$; hence

$$\begin{aligned} \varphi(x, r) &= x, \\ \varphi(x, 2r) &= \varphi(\varphi(x, r), r) = \varphi(x, r) = x, \\ &\dots \\ \varphi(x, ur) &= \varphi(\varphi(x, (u-1)r), r) = \varphi(x, r) = x, \\ \varphi(x, t^*) &= \varphi(x, ur + v) = \varphi(\varphi(x, ur), v) = \varphi(x, v) = x, \end{aligned}$$

this means $t^* \in H$. \square

In the set of permitted states of an autonomous continuous automaton, we introduce the (binary) relation σ as follows: $\sigma(x, y)$ exactly if there exist two non-negative numbers t_1, t_2 such that $\varphi(x, t_1) = y$ and $\varphi(y, t_2) = x$. It is obvious that σ is reflexive, symmetric and transitive, consequently the set of permitted states splits into equivalence classes modulo σ (called σ -classes). A σ -class is said *trivial* if it consists of one element only.

If $\varphi(x, t)$ (is meaningful and) equals x for every non-negative t , then x is called a *steady state*. Any steady state forms a trivial σ -class (but the trivial σ -classes are not exhausted in this manner). If the σ -class containing x is non-trivial, then we say that x is a *properly periodic state*¹² and its *period* is p_x . x is *periodic* if it is either steady or properly periodic.

The behaviour of an initial automaton whose initial state x_0 is steady is trivial. If x_0 is properly periodic, then the behaviour may be derived from our next statement:

Proposition 13. *Let K be a non-trivial σ -class of an autonomous continuous automaton. Then the following five assertions hold:*

- (I) *(the period p_x is common in K , i.e.) there exists a positive number p_K such that $p_x = p_K$ for every $x \in K$,*
- (II) *whenever $x \in K$ and $t^* \geq 0$, then $\varphi(x, t^*)$ (exists and) belongs to K ,*
- (III) *whenever x and y are arbitrary elements of K , then there exists one and only one instant t fulfilling both $0 \leq t < p_K$ and $\varphi(x, t) = y$,*
- (IV) *whenever $x \in K$ and $t^* \geq p_K$, then $\varphi(x, t^*) = \varphi(x, v)$ where v is the single number determined by the inequalities $0 \leq v < p_K$ and the condition that $(t^* - v)/p_K$ is an integer,*
- (V) *the cardinality of K is continuum.*

Proof. (I) Choose two arbitrary elements x, y of K . Since $\varphi(x, t_1) = y$ is satisfiable with some t_1 , we have

$$\varphi(y, p_x) = \varphi(\varphi(x, t_1), p_x) = \varphi(x, t_1 + p_x) = \varphi(\varphi(x, p_x), t_1) = \varphi(x, t_1) = y$$

hence $p_y \leq p_x$. A symmetrical inference shows that $p_x \leq p_y$. Hence p_x is the same for every choice of x in K .

¹² Proposition 13 will justify this terminology.

(II) Let u, v be two numbers such that $t^* = up_x + v$, $0 \leq v < p_x$ and u is an integer. An easy induction shows that $\varphi(x, up_x)$ (exists and) equals x for every u (similarly to the final part of the proof of Proposition 12). Moreover,

$$\varphi(x, t^*) = \varphi(x, up_x + v) = \varphi(\varphi(x, up_x), v) = \varphi(x, v)$$

(where $\varphi(x, v)$ is necessarily meaningful!) and

$$\varphi(\varphi(x, t^*), p_x - v) = \varphi(\varphi(x, v), p_x - v) = \varphi(x, v + p_x - v) = \varphi(x, p_x) = x,$$

consequently, x and $\varphi(x, t^*)$ are in the same σ -class.

(III) We have $\varphi(x, t^*) = y$ with some $t^* (\geq 0)$. The number v , seen in the preceding section of the proof, is a convenient v . We are going to show the unicity of t . Suppose (contrarily to this) that $\varphi(x, t) = \varphi(x, t') = y$ where $0 \leq t < t' < p_K$; hence

$$\begin{aligned} \varphi(x, p_K + t - t') &= \varphi(\varphi(x, t), p_K - t') = \\ &= \varphi(\varphi(x, t'), p_K - t') = \varphi(x, t' + p_K - t') = \varphi(x, p_K) = x, \end{aligned}$$

this equality and the obvious inequalities

$$0 < (p_K - t') < p_K + t - t' < p_K$$

contradict the definition of p_x .

(IV) This assertion was already verified in the proof of (II).

(V) If x is fixed, then (III) establishes a one-to-one mapping between the elements (denoted by y) of K and the numbers being in the interval $[0, p_K)$. \square

Now we turn to the behaviour of an automaton when it starts with a non-periodic state.

Proposition 14. *Let an autonomous continuous automaton be considered. Suppose that a permitted state x of it is not periodic. Then exactly one of the subsequent two assertions is valid:*

(A) *The states $\varphi(x, t)$ are pairwise different as far as they are defined (the bound b_x may or may not exist).*

(B) *There is a number $c_x (\geq 0)$ such that*

(i) *for all choices of t such that $t > c_x$, the states $\varphi(x, t)$ are (defined and) periodic and moreover, they belong to a common σ -class K ,*

(ii) *for the choices of t such that $0 \leq t < c_x$, the states $\varphi(x, t)$ are non-periodic and pairwise different, moreover, any of them differs from $\varphi(x, c_x)$, and*

(iii) *if ($c_x > 0$ and) $\varphi(x, c_x)$ is a periodic state, then it belongs to the class K mentioned in (i).*

Proof. Assume that (A) does not hold, i.e. there are two instants t_1, t_2 such that $(0 <) t_1 < t_2$ and $\varphi(x, t_1) = \varphi(x, t_2)$. We want to show that all statements of (B) are true.

Case 1: there is a t' such that $t_1 < t' < t_2$ and $\varphi(x, t') \neq \varphi(x, t_1)$. In sense of the formulae $\varphi(\varphi(x, t_1), t' - t_1) = \varphi(x, t')$, $\varphi(\varphi(x, t'), t_2 - t') = \varphi(x, t_2) = \varphi(x, t_1)$, $\varphi(x, t_1)$ and $\varphi(x, t')$ are in the same σ -class K , hence they are properly periodic. Denote by $J (\neq \emptyset)$ the set of all instants t^* fulfilling $\varphi(x, t^*) \in K$. Proposition 13,

(II) implies that J is an interval of form (c_x, ∞) or $[c_x, \infty)$ where c_x is the infimum of J . The statements (i), (iii) are obvious, (ii) can be verified easily by indirect method (namely, supposing that t_1, t_2 may be chosen so that $t_1 < t_2 < c_x$, we get a contradiction with the definition of c_x).

Case 2: $\varphi(x, t') = \varphi(x, t_1)$ whenever t' is between t_1 and t_2 . It is clear that $\varphi(x, t_1)$ is a steady state. If we denote now by J the set of instants t^* fulfilling $\varphi(x, t^*) = \varphi(x, t_1)$, then J and its infimum c_x will again satisfy (i), (ii), (iii). \square

A (non-periodic, permitted) state is called *aperiodic* or *pre-periodic* if it satisfies assertion (A) or assertion (B) of Proposition 14, respectively. The number c_x is called the *length* of the pre-period. An aperiodic state x is called *bounded aperiodic* or *boundless aperiodic* depending on the existence of the bound b_x . Table 1 shows the hierarchy of the notions introduced for states.

PROBLEM 9. Analyze how the periodicity properties are modified if some of the conditions (i)—(vii) is replaced by a weaker one.

(E.g. we can suppose, instead of (ii), that t varies on a totally ordered set T , the cardinality of T may differ from the continuum.)

<div style="display: flex; justify-content: space-between; align-items: center;"> <div style="display: flex; flex-direction: column; gap: 5px;"> properly periodic steady </div> <div style="font-size: 2em;">}</div> periodic </div>	<div style="display: flex; flex-direction: column; gap: 5px;"> pre-periodic boundless aperiodic bounded aperiodic </div>	<div style="display: flex; flex-direction: column; gap: 5px;"> aperiodic </div>	<div style="display: flex; flex-direction: column; gap: 5px;"> non-periodic </div>	<div style="display: flex; flex-direction: column; gap: 5px;"> permitted </div>
--	---	--	---	--

Table 1

Remarks. In this § we have considered automata whose functioning starts with an *instantaneous* initial state. The somewhat modified notion when the behaviour in an interval of *positive* duration is regarded to be the starting condition has been studied by Konikowska [15].

It is worthy of mention that the behaviour of autonomous systems of differential equations has certain similarities to the above results (see [17], § 15).

§ 9.

By a *network* we understand a triple (G, Σ, φ) such that

- (1) G is a finite directed graph whose vertices are denoted by P_1, P_2, \dots, P_n ,
- (2) Σ is a subset of the set of real numbers,
- (3) the set of all mappings x of the set $\{P_1, P_2, \dots, P_n\}$ into Σ is denoted by X , and
- (4) φ is an autonomous continuous automaton, the set of states of φ is X , the function φ is given in terms of the graph structure of G .

By virtue of the above definition a state of the network in question is denoted as a vector

$$(x(P_1), x(P_2), \dots, x(P_n)). \quad (9.1)$$

For the sake of convenience, we write the simpler notation

$$(x_1, x_2, \dots, x_n) \quad (9.2)$$

instead of the vector (9.1) and we identify the mapping x to (9.2).

Thus $\varphi(x, t)$ may be decomposed into n functions

$$\alpha_1(x, t), \alpha_2(x, t), \dots, \alpha_n(x, t) \quad (9.2)$$

where $\alpha_i(x, t)$ is the i -th component of the vectorial form of the state¹³ $\varphi(x, t)$.

This means that if x_i is the state of the vertex P_i of the network at some instant t_0 (for any i , $1 \leq i \leq n$) and we use the symbol x for the sequence (x_1, x_2, \dots, x_n) , then $\alpha_i(x, t)$ denotes the state of some P_i at $t_0 + t$.

§ 10.

Now we define a particular type of networks which will be called *simple* networks in the sequel¹⁴. This notion arises (by abstraction) from the networks studied in Chapter 2 of [14]. Let G be a graph (without loops and multiple edges). Let Σ be the closed interval $[0, 1]$ (i.e. the set of real numbers y fulfilling $0 \leq y \leq 1$). Let the positive number τ , called *recovery time* (of the vertices), be characteristic for the network (τ is the same for each vertex). The function φ is determined by the subsequent four rules (in which $\{y\}$ is defined by

$$\{y\} = \begin{cases} y & \text{if } 0 \leq y < 1 \\ 0 & \text{if } y = 1 \end{cases}$$

and ϱ_x denotes the maximum of the values $\{x_1\}, \{x_2\}, \dots, \{x_n\}$):

(1) a state x is permitted if and only if the equality $x_i = 1$ and the existence of the edge $\overline{P_i P_j}$ imply $x_j = 0$ (for any i, j where $1 \leq i \leq n$, $1 \leq j \leq n$),

(2) if x is a permitted state and $0 < t < (1 - \varrho_x)\tau$, then we define

$$\varphi(x, t) = (z_1, z_2, \dots, z_n)$$

¹³ The functions $\alpha_i(x, t)$ are *not* autonomous continuous automata. (Indeed, let us recall the definition of autonomous continuous automata in § 8. If the set of states of the entire network is denoted by X , then (iii) is not satisfied; if X denotes the state of a single vertex, then (i) is not fulfilled.) — We avoid the notation of type $\alpha_i(x_i, t)$ since it lacks to be a function (because there are interactions among the vertices; hence $\alpha_i(x, t)$ may depend on each of x_1, x_2, \dots, x_n , not only on x_i).

¹⁴ The reader who is interested in this subject may find a more detailed explanation in [2] (mainly in Section 3). A short summary of the definition is contained also at the beginning of § 6 of [3].

in the following manner:

$$z_i = \begin{cases} 1 & \text{if } x_i = 1 \\ 0 & \text{if there is a } j \text{ such that } \overrightarrow{P_j P_i} \text{ exists and } x_j = 1, \\ x_i + \frac{t}{\tau} & \text{otherwise,} \end{cases}$$

(3) if x is a permitted state, $q_x > 0$ and there are two subscripts i, j such that $\overrightarrow{P_i P_j}$ exists and $x_i = x_j = q_x$, then $\varphi(x, (1 - q_x)\tau)$ is undefined,

(4) if x is a permitted state, $q_x > 0$ and the assumption in (3) does not hold, then we define

$$\varphi(x, (1 - q_x)\tau) = (z_1, z_2, \dots, z_n)$$

in the following manner

$$z_i = \begin{cases} 0 & \text{if there is a } j \text{ such that } \overrightarrow{P_j P_i} \text{ exists and} \\ & \text{one of the equalities } x_j = q_x, x_j = 1 \text{ is true,} \\ \min(1, x_i + 1 - q_x) & \text{otherwise}^{15}. \end{cases}$$

It can be verified that we have defined an autonomous continuous automaton φ (among others, either the value $\varphi(x, t)$ can be determined or the fact that $\varphi(x, t)$ is not defined can be stated consistently by successive application of (1)–(4) for an arbitrary choice of the state x and the non-negative instant t).

Our next aim is to clear up the intuitive basis of the above definition. The vertices of the network represent (idealized) neurons, the edges of the network are inhibitory connections. Each neuron is (at an instant) either in inhibited state or in firing state (denoted by 0, 1, resp.) or in the so-named recovery state¹⁶ (being between these extremities). The inhibition is understood in such a manner that if a neuron P is in firing state and the edge \overrightarrow{PQ} exists, then Q is in inhibited state. There is a permanent "background effect" manifesting itself in such a way that a neuron, being in recovery state, strives to be firing (unless, of course, it will be inhibited by another state in the meantime), and a firing neuron remains in firing state (till it gets an inhibition). The inhibition is produced instantaneously (and, more precisely, right-continuously), the duration of a full recovery phase (beginning with an inhibited state and leading to a firing state) is τ , this duration does not depend on which of the neurons is considered. If two neurons, connected with each other, reach the firing state at the same instant, then the behaviour of the network is studied no more. Let the state of a neuron, being in recovery state at an instant t , be considered; if the neuron is in recovery phase since a duration of length t_0 , then its state is denoted by t_0/τ (clearly $0 < t_0/\tau < 1$), hence the firing state 1 will be reached continuously at the instant $t + \tau - t_0$ (except when the recovery phase is interrupted by a new inhibition) (see Fig. 4).

¹⁵ The definition of q_x implies that $x_i + 1 - q_x > 1$ if and only if $x_i = 1$.

¹⁶ The terminology "recovery" occurs in another sense in the literature, too, than its meaning in the sequel.

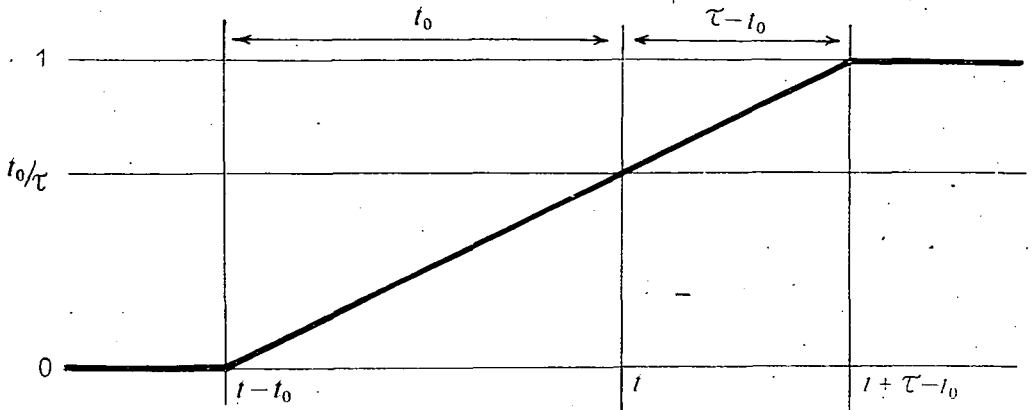


Fig. 4

It is shown in Sections 4—5 of [2] that the functioning of such a network happens in *discrete* steps substantially, and the essential instants can be calculated from the initial state. It follows from the treatment that no boundless aperiodic state occurs in this type of networks. A matrix

$$\begin{pmatrix} \Psi_0^0 & \Psi_1^0 & \dots & \Psi_{q+1}^0 \\ \Psi_0^1 & \Psi_1^1 & \dots & \Psi_{q+1}^1 \\ \dots & \dots & \dots & \dots \\ \Psi_0^m & \Psi_1^m & \dots & \Psi_{q+1}^m \\ \dots & \dots & \dots & \dots \end{pmatrix} \quad (10.1)$$

(having an infinity of rows) is there considered whose entries are sets of vertices (corresponding to the smaller or larger values of $\alpha_i(x, T_m)$ where x is the initial state and T_m is the m -th essential instant), and rules are stated how the entries of the m -th row may be expressed in terms of the entries of the $(m-1)$ -th row. The following three problems are mentioned in the last section of [2]:

PROBLEM 10. Express the entries $\Psi_0^m, \Psi_1^m, \dots, \Psi_{q+1}^m$ of (10.1) in terms of the entries $\Psi_0^0, \Psi_1^0, \dots, \Psi_{q+1}^0$ by formulae which are closed as far as possible.

PROBLEM 11. Describe the periodicity properties of a network (starting with an arbitrary initial state) by use of the matrix (10.1).

PROBLEM 12. Characterize the graphs G possessing the stability property that whenever x and y are periodic or pre-periodic states, then there exist t and t' such that $\varphi(x, t) = \varphi(y, t')$.

(Obviously, Problems 10 and 11 are closely related to each other.)

§ 11.

It seems that from various considerations (belonging not to mathematics but to more or less experimental sciences) it is imaginable to derive various network types. In [2] it was elaborated only in one special case how some concrete type of networks can be introduced and analyzed.

Among the diversity of possibilities, we turn now to the network type suggested by Chapter 3 of [14] where they are called "complex networks". We are going to give a definition of these networks that uses mathematical tools (analogously to how the network notion appearing in Chapter 2 of [14] was abstractly defined in Section 3 of [2]). (The motivation will be given at the end of the §.) The definition consists of four parts.

(I) The graph G contains $2n$ vertices which are presented in a matrix form:

$$\begin{pmatrix} P_1 & P_2 & \dots & P_n \\ Q_1 & Q_2 & \dots & Q_n \end{pmatrix} \tag{11.1}$$

The n edges $\overrightarrow{P_1 Q_1}, \overrightarrow{P_2 Q_2}, \dots, \overrightarrow{P_n Q_n}$ are always present in G , every other edge of G starts from some Q_i .

(II) Σ consists of the real numbers a satisfying either $0 \leq a \leq 1$ or $a = 2$.

(III) A state

$$x = \begin{pmatrix} x(P_1) & x(P_2) & \dots & x(P_n) \\ x(Q_1) & x(Q_2) & \dots & x(Q_n) \end{pmatrix} = \begin{pmatrix} x_1 & x_2 & \dots & x_n \\ z_1 & z_1 & \dots & z_n \end{pmatrix} \tag{11.2}$$

is permitted precisely when

- (a) $x_1 \neq 2, x_2 \neq 2, \dots, x_n \neq 2,$
- (b) $x_i = 1$ implies $z_i \neq 1,$
- (c) $z_i = 2$ implies $x_i = 1,$
- (d) $z_i = 2$ and the existence of $\overrightarrow{Q_i P_j}$ imply $x_j = 0,$ and
- (e) $z_i = 2$ and the existence of $\overrightarrow{Q_i Q_j}$ imply $z_j = 0.$

Before exposing the final part of the definition, we agree that two (fixed) positive real numbers τ_x, τ_z are characteristic for the activity of the network, and we introduce some notations (concerning (11.2)). Let σ be the minimum of the positive numbers among the $2n$ quantities

$$\tau_x(1-x_1), \tau_x(1-x_2), \dots, \tau_x(1-x_n), \tau_z(1-z_1), \tau_z(1-z_2), \dots, \tau_z(1-z_n).$$

Γ^p is the set of subscripts i fulfilling $x_i = 1$. $\Gamma^{a,2}$ and $\Gamma^{a,1}$ are the sets of i 's satisfying $z_i = 2$ or $z_i = 1$, respectively. Δ^p and Δ^q are the sets of i 's for which $x_i = 1 - \frac{\sigma}{\tau_x}$ or

$z_i = 1 - \frac{\sigma}{\tau_z}$ hold, respectively.¹⁷ If Ψ is an arbitrary subset of $\{1, 2, \dots, n\}$, then

let $\chi_p(\Psi)$ be the set of those P_i 's for which $\overrightarrow{Q_j P_i}$ exists in G with a suitable $j (\in \Psi)$. The set $\chi_q(\Psi)$ (consisting of some Q_i 's) is similarly defined (with $\overrightarrow{Q_j Q_i}$).

¹⁷ If τ_x or τ_z equals σ , then the definitions of Δ^p or Δ^q must be somewhat modified. This may be left to the reader (for an analogy, see the definition of Δ_k and Footnote 4 in [2]).

(IV) Now define the values of the entries of the matrix

$$\varphi(x, t) = \begin{pmatrix} \alpha_1(x_1, t) & \alpha_2(x_2, t) & \dots & \alpha_n(x_n, t) \\ \gamma_1(z_1, t) & \gamma_2(z_2, t) & \dots & \gamma_n(z_n, t) \end{pmatrix}$$

where x is the same as in (11.2) and t satisfies (separately) $0 < t < \sigma$ or $t = \sigma$ in the following way:

[a] if $t < \sigma$, then

$$\alpha_i(x_i, t) = \begin{cases} 0 & \text{if } P_i \in \chi_p(\Gamma^{q,2}) \\ 1 & \text{if } i \in \Gamma^p \\ x_i + \frac{t}{\tau_x} & \text{otherwise,} \end{cases}$$

$$\gamma_i(z_i, t) = \begin{cases} 0 & \text{if } Q_i \in \chi_q(\Gamma^{q,2}) \\ 1 & \text{if } i \in \Gamma^{q,1} \\ 2 & \text{if } i \in \Gamma^{q,2} \\ z_i + \frac{t}{\tau_z} & \text{otherwise,} \end{cases}$$

[b] for $t = \sigma$:

$$\alpha_i(x_i, \sigma) = \begin{cases} 0 & \text{if } P_i \in \chi_p(\Delta^p \cap \Delta^q) \\ 1 & \text{if } i \in \Gamma^p \text{ and } P_i \notin \chi_p(\Delta^p \cap \Delta^q) \\ 1 & \text{if } i \in \Delta^p \\ x_i + \frac{\sigma}{\tau_x} & \text{otherwise} \end{cases}$$

$$\gamma_i(z_i, \sigma) = \begin{cases} 0 & \text{if } Q_i \in \chi_q(\Delta^p \cap \Delta^q) \\ 1 & \text{if } i \in \Gamma^{q,1} - (\Gamma^p \cup \Delta^p) \text{ and} \\ & Q_i \notin \chi_q(\Delta^p \cap \Delta^q) \\ 2 & \text{if } i \in \Delta^q \cap (\Gamma^p \cup \Delta^p) \\ 2 & \text{if } i \in (\Gamma^{q,2} \cup \Gamma^{q,1}) \cap (\Gamma^p \cup \Delta^p) \text{ and} \\ & Q_i \notin \chi_q(\Delta^p \cap \Delta^q) \\ z_i + \frac{\sigma}{\tau_z} & \text{otherwise.} \end{cases}$$

It may happen that this definition of $\alpha_i(x_i, \sigma)$ or $\gamma_i(z_i, \sigma)$ is not consistent (because the first and third conditions in both definitions do not exclude each other in general). If such a contradiction arises, then we do not define $\varphi(x, t)$ for the instants that are $\cong \sigma$.

If the values $\alpha_i(x_i, \sigma)$ and $\gamma_i(z_i, \sigma)$ are meaningful, then the definition of φ can be continued such that 0 is replaced by σ and some instant $\sigma' (> \sigma)$ will play the role of σ . This may be continued piece-wise till the infinity unless a contradiction is sometimes produced (in the manner seen above).

PROBLEM 13. Let the network type introduced in this § be studied, let the analogies and dissimilarities to [2] be discovered.

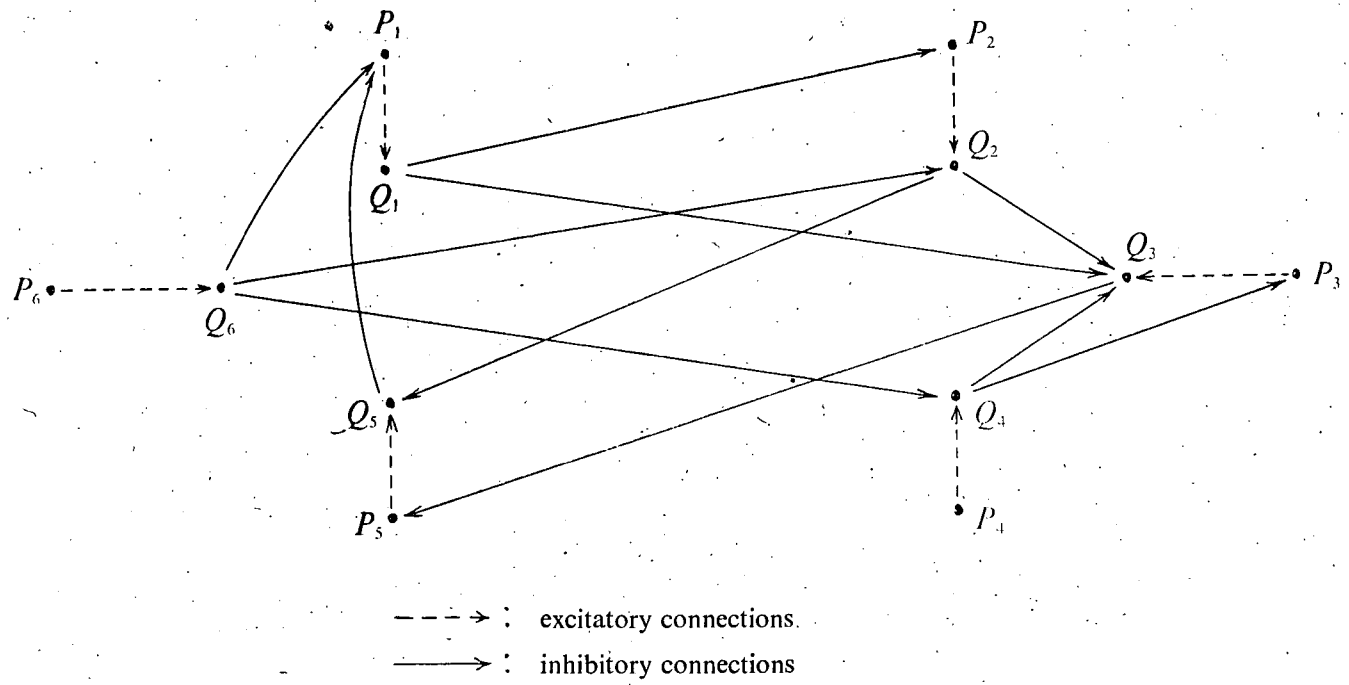


Fig. 5. A complex network.

Since the above definition of complex networks is awfully intricate, the reader may expect eagerly the usual elucidating considerations. The present notion is a modified (and, we can say, improved) version of the simple networks (in § 10). There are two types of neurons (denoted by P 's and Q 's with subscripts, resp.) and the neurons constitute pairs of form (P_i, Q_i) . The edges of form $P_i Q_i$ express *excitatory* connections, the remaining ones (each starting from a Q_i) express inhibiting ones. The inhibited state is denoted by 0 and the recovery state is by numbers in the open interval $(0, 1)$ (without any change). In contrast to the simple networks, now two kinds of firing state are defined for the vertices Q_i , they are denoted by the numbers¹⁸ 1 and 2 (for the P_i 's only the firing state 1 is permitted). A neuron Q_i is in the state 2 if and only if both of Q_i and P_i have finished a recovery phase (and they did not get inhibition in the meantime). If Q_i is in the state 1, then Q_i does not really produce an inhibiting effect (but it is ready to produce the effect instantaneously when it gets an excitation from its pair P_i). The "background effect" acts similarly to the simple networks (but this effect alone is unable to produce the state 2 for a Q_i). The recovery durations τ_x and τ_z (for the P_i 's and Q_i 's, resp.) may differ from each other. The detailed prescriptions in (IV) are elaborated in analogy with Section 3 of [2]. Finally we specify the meaning of some formulae in (IV). $\chi_p(\Gamma^{q,2})$ is the set of P_i 's being inhibited by a Q_i at the initial instant 0. $\chi_p(\Delta^p \cap \Delta^q)$ is the set of P_i 's for which an inhibition sets in at the instant σ . $\Gamma^{q,1} - (\Gamma^p \cup \Delta^p)$ is the set of subscripts i such that (1) the state of Q_i was 1 during the interval $[0, \sigma)$ and (2) the state of P_i does not reach 1 at σ . $\Delta^q \cap (\Gamma^p \cup \Delta^p)$ is the set of i 's such that (1) the state of Q_i converges to 1 (from below) if t approximates σ (from the left) and (2) the state of P_i reaches 1 at σ . $(\Gamma^{q,2} \cup \Gamma^{q,1}) \cap (\Gamma^p \cup \Delta^p)$ is the set of i 's such that (1) the state of Q_i is 1 or 2 during $[0, \sigma)$ and (2) the state of P_i reaches 1 at σ .

III. On the interconnections between structure and function

§ 12.

The exact mathematical treatment of the behaviour of networks dealt with in Chapter 2 of [14] has been done in the article [7]. We applied in [7] the considerations of [2] to the more particular type of networks whose structure is $G(n; 1, 2, \dots, k)$ with some n and k where $1 \leq k < n$, $n \geq 3$ (cf. § 3 and § 10 of this paper); this specialization enables us to deduce more explicit assertions on the behaviour in comparison to the case when (in [2]) we did not restrict ourselves to any special graph structure.

We have introduced in [7] the notion of regular state in terms of certain equalities and inequalities between the values x_1, x_2, \dots, x_n . The main consequences are: any regular state is periodic and its period is a divisor of the number¹⁹ $\frac{\tau[n, k+1]}{k+1}$ (if it is properly periodic),

any non-regular state is either pre-periodic or bounded aperiodic and — respectively to these cases — the corresponding number c_x or b_x does not exceed 2τ .

§§ 6—8 of [3] have been devoted to an extension of the results of [7] to the networks the structure of which belongs to the graph class $\mathfrak{A}(C_\beta)$ where C_β is the family of (β) -graphs (see § 1 and § 5). It was proved that — after a suitable defini-

¹⁸ The occurrence of the isolated number 2 in (II) means that we have given up certain continuity properties of the mathematical model.

¹⁹ $[a, b]$ denotes here the least common multiple of a and b . We say that a is a divisor of b if $ac = b$ with some positive integer c (a and b are not necessarily integers). The word "period" is now meant as it was defined in § 8. (This is the same as "smallest period" in the terminology of the previous articles. Other changes in the terminology are that we say now "periodic" and "non-periodic" instead of "cyclic" and "acyclic", resp.)

tion of regularity — each regular state is periodic; I did not succeed, however, in showing the expected converse of this statement. Let therefore the conjecture that terminates the article [3] be recalled as follows:

PROBLEM 14. *Decide whether or not there exists a properly periodic non-regular state of a network lying in $\mathfrak{A}(C_\beta)$.*

The next question is of comprehensive nature, it proposes further investigations in analogy with [7] (the activity of a network is thought again as it was studied in [2]):

PROBLEM 15. *For network classes corresponding to various graph types affected in Chapter I of this paper, determine the sets of periodic states and the other periodicity properties.*

§ 13.

Let henceforward the network type of [2] be considered. We mention a mathematical formulation of the question how rapidly the effects can be propagated in a network.

Whenever a network and a permitted state $x=(x_1, x_2, \dots, x_n)$ of it is given and the edge $\overrightarrow{P_j P_i}$ exists, then we call $\overrightarrow{P_j P_i}$ a red edge or a green edge accordingly to which of $x_j < x_i, x_j > x_i$ is true²⁰. For emphasizing the role of x we can speak of x -red and x -green edges.

We say that the independence statement $I(r, s, t)$ holds if

$$\alpha_i(x, t) = \alpha_i(y, t)$$

is true for every possible choice of G, i, x, y where the occurring symbols have the following meanings:

- (i) r and s are non-negative integers, at least one of them is positive,
- (ii) t is a positive real number,
- (iii) (G, Σ, φ) is a network (as in § 10),
- (iv) the number of vertices of G is denoted by n ,
- (v) i is an integer such that $1 \leq i \leq n$,
- (vi) $x=(x_1, x_2, \dots, x_n)$ and $y=(y_1, y_2, \dots, y_n)$ are states of (G, Σ, φ) , each edge of G is supposed to be red or green concerning any of x, y ,
- (vii) any integer j ($1 \leq j \leq n$) fulfils the condition: either $x_j = y_j$ or each path from P_j to P_i (in G) contains at least r x -red edges, at least r y -red ones, at least s x -green ones and at least s y -green ones.

Conditions (i), (vii) imply $x_i = y_i$ since $x_j \neq y_j$ is possible only if the number of edges of an arbitrary path from P_j to P_i is at least $r+s (>0)$.

We define $\pi(r, s)$ as the largest number u (possibly ∞) possessing the following property: the independence statement $I(r, s, t)$ is true for every t such that $0 < t < ut$.

²⁰ The edge is not coloured if $x_i = x_j$.

PROBLEM 16. Let the function $\pi(r, s)$ be studied.

In the particular case $s=0$, I conjecture $\pi(r, 0) = \left\lfloor \frac{r-1}{2} \right\rfloor$ ($[a]$ denotes here the integer fulfilling $a-1 < [a] \leq a$).

Example. Let the graph with the four vertices P_1, P_2, P_3, P_4 and three edges $\overrightarrow{P_1 P_2}, \overrightarrow{P_2 P_3}, \overrightarrow{P_3 P_4}$ be considered with the initial states $x=(0.96, 0.97, 0.98, 0.99)$ and $y=(0.95, 0.97, 0.98, 0.99)$. An easy discussion shows that

$$(1) \quad \begin{aligned} \alpha_2(x, t) &= \alpha_2(y, t) \quad \text{if } t < 0.04\tau \quad \text{but} \\ \alpha_2(x, 0.04\tau) &= 0 \neq 1 = \alpha_2(y, 0.04\tau), \end{aligned}$$

$$(2) \quad \begin{aligned} \alpha_3(x, t) &= \alpha_3(y, t) \quad \text{if } t \leq 0.04\tau \quad \text{but e.g.} \\ \alpha_3(x, 0.05\tau) &= 0.01 \neq 0 = \alpha_3(y, 0.05\tau), \end{aligned}$$

$$(3) \quad \begin{aligned} \alpha_4(x, t) &= \alpha_4(y, t) \quad \text{if } t < 1.04\tau \quad \text{but} \\ \alpha_4(x, 1.04\tau) &= 0 \neq 1 = \alpha_4(y, 1.04\tau). \end{aligned}$$

Hence this example implies $\pi(1, 0) \leq 0.04$, $\pi(2, 0) \leq 0.04$, $\pi(3, 0) \leq 1.04$.

Remark. It may seem to be curious that red and green edges were distinguished in the above definition of π . Now we want to explain why this was done. In fact, one can introduce $I^*(q, t)$ and $\pi^*(q)$ in an analogous (but simpler) manner; this function π^* is, however, identically zero. E.g. the discussion of the network having the edges $\overrightarrow{P_1 P_2}, \overrightarrow{P_2 P_3}, \overrightarrow{P_3 P_4}, \overrightarrow{P_4 P_5}$ with the initial states $x=(0, 0.98, 0.97, 0.96, 0.95)$ and $y=(0.99, 0.98, 0.97, 0.96, 0.95)$ shows $\pi^*(4) \leq 0.04$.

§ 14.

Let the so-named complex networks be considered the formalized definition of which was contained in § 11.

Suppose that H_p and H_q are non-empty finite sets of positive integers such that

$$H_p \cup H_q = \{1, 2, 3, \dots, |H_p| + |H_q|\}.$$

(This equality implies $H_p \cap H_q = \emptyset$). Let n, k fulfil $n > k = |H_p| + |H_q|$; we denote by $G(n; H_p, H_q)$ the graph whose vertices are $P_1, Q_1, P_2, Q_2, \dots, P_n, Q_n$ and the edges of which are determined in the following way:

$\overrightarrow{P_i Q_i}$ exists for every i ($1 \leq i \leq n$),

$\overrightarrow{Q_i P_j}$ exists precisely when there is an $h (\in H_p)$ such that $i - j \equiv h \pmod{n}$,

$\overrightarrow{Q_i Q_j}$ exists precisely when there is an $h (\in H_q)$ such that $i - j \equiv h \pmod{n}$.

Chapter 3 of [14] gives a suggestion for raising the following question:

PROBLEM 17. Let the complex networks (in sense of [14]) built up over the class of graphs expressible in the form $G(n; H_p, H_q)$ be studied (in a manner analogous to [7]).

IV. On the stochastic behaviour of networks

§ 15.

In the network type considered in § 10, the recovery time τ was supposed to be (common for the vertices and) strictly determined. It may be a better simulation of real processes if the value of τ is randomly chosen (at every occasion when a recovery phase takes place) according to some probabilistic distribution. E.g. the following question may be raised:

PROBLEM 18. *Let a (stochastic) analogon of the investigations mentioned in § 10 and § 12 be given when τ is a logarithmically normally distributed stochastic variable (i.e. $\tau = e^{\tau'}$ where τ' is distributed normally).*

§ 16.

Another possibility for probabilistic considerations arises if (τ is deterministic but) the initial state of a network is not fixed. Namely, let the case be considered when

(a) the components of the initial state are numbers chosen randomly (e.g. in sense of the uniform distribution) between 0 and 1,

(b) the class of all (logically possible) manners of behaviour is partitioned to some subclasses K_1, K_2, \dots by virtue of some simple properties, and

(c) we are interested in the probability $P(K_i)$ of the event that an initial state will lead to a behaviour belonging to K_i .

In § 5 of [7] we have proposed a problem of this type when the graph structure of the networks is $G(n; 1)$ and a behaviour is defined to belong to the class K_i if it leads to a regular state with precisely $i (\leq n/2)$ vertices being in the maximal state 1.

In §§ 4—5 of the paper [6] an attempt was made for showing how a particular problem, being similar to the mentioned type, can be studied. The following question was discussed:

(i) we consider the networks built up on finite trees with a distinguished vertex (called *root* of the tree) such that every edge is directed towards the root,

(ii) the components of the initial state of a network are chosen randomly (like in (a)),

(iii) we separate five types of the behaviour of the networks (each starting from an initial state) according to which of the following statements are fulfilled:²¹

the state of the root is 1 somewhere in the open interval $(0, \tau)$,

the state of the root is 0 at τ ,

the state of the root is 1 at τ ,

(iv) the probabilities of the occurrence of the types are calculated such that the number of vertices of the tree is fixed and the trees (having this size) are chosen equiprobably with respect to an isomorphism notion.

²¹ These three statements can be combined with each other in eight manners. From among these (logically imaginable) cases, two ones are impossible, a third one is of zero probability.

§ 17.

In the last section of this paper a glance is thrown at the problem of genesis of networks, i.e. how a graph (underlying a network) may develop in a stochastic manner. In the article [11] the following version of this question is thoroughly studied (concerning non-directed graphs):

at the beginning of the process there are given n isolated vertices (this set of vertices remains unchanged),

after i steps we start with a graph having (the n vertices and) i edges from among the $\frac{n(n-1)}{2}$ possible ones, the $(i+1)$ -th step is that we draw a further edge e such

that e joins one of the $\frac{n(n-1)}{2} - i$ non-adjacent vertex pairs and is chosen equi-probably,

the procedure terminates after k steps.

Erdős and Rényi have determined in [11] which graphs may be typically formed in this manner (depending on the order of magnitude of k). It seems that these typical graphs are quite dissimilar from the graphs belonging to the classes dealt with in our Chapter I. Therefore, if one shares the opinion that the elements of our graph classes are particularly able as carriers of networks producing reasonable activities, then he must seek other principles in addition to the above principle of „inserting a new edge” in sense of Erdős and Rényi.

Such an additional principle may be that an edge ceases to exist under certain circumstances. If some condition, implying that edges are destroyed, is fixed, then one can study e.g. what happens typically when he starts with a complete graph (with n vertices and all the possible $\frac{n(n-1)}{2}$ edges, being the edges oriented randomly and independently of each other) and applies the “edge-destroying” principle in k steps. Of course, the principles of inserting and destroying may also be combined with each other.

Let us return again to the manner of functioning introduced in § 10. A concrete possibility how an edge may cease is illustrated in the following example. Let a network have four vertices P_1, P_2, P_3, P_4 , four edges $\overrightarrow{P_1P_2}, \overrightarrow{P_2P_3}, \overrightarrow{P_2P_4}, \overrightarrow{P_3P_4}$ and the initial state $(0.6, 0.7, 0.8, 0.9)$. A discussion of the activity of this network shows that it is defined only for the instants t smaller than 1.4τ . When the instant 1.4τ is approximated, then both of $\alpha_3(t)$ and $\alpha_4(t)$ converge to 1; this fact and the existence of the edge $\overrightarrow{P_3P_4}$ imply that the working of this network remains undefined if $t \geq 1.4\tau$. It seems advisable to supplement the network definition by demanding that an edge e is deleted at such an instant when both vertices incident to e reach the value 1 simultaneously.

PROBLEM 19. *Let the activity of a network be understood as in § 10. Which typical graph structures (depending on the order of magnitude of k) arise if we start with a complete graph with n vertices and we apply the edge-destroying principle, mentioned above, till when k edges had been deleted?*

It is expectable that the stochastic considerations proposed in § 16 are closely related to the study of Problem 19.

I wish to express my thanks to Dr. E. LÁBOS and Dr. L. LOVÁSZ (the official referees of this article) for a number of useful advices.

**О некоторых открытых проблемах прикладной теории автоматов
и теории графов (возбуждаемых математическим моделированием неких
нейрональных сетей)**

Статья [14] U. Kling-а и Gy. Székely-а брала инициативу к математическому моделированию строения и функционированию некоторых нейрональных сетей, см. работы [2], [7], [3] и т. д. В настоящей статье дается краткий обзор этих исследований и специфицируется ряд дальнейших нерешенных математических вопросов. В первой из четырех глав работы содержатся проблемы относительно конечных ориентированных графов.

MATHEMATICAL INSTITUTE OF THE
HUNGARIAN ACADEMY OF SCIENCES
1053 BUDAPEST, HUNGARY
REÁLTANODA U. 13.

References

- [1] ÁDÁM, A., Research problem 2—10 (Isomorphism problem for a special class of graphs), *J. Combinatorial Theory*, v. 2, 1967, p. 393.
- [2] ÁDÁM, A., Simulation of rhythmic nervous activities, II. (Mathematical models for the function of networks with cyclic inhibitions), *Kybernetik*, v. 5, 1968, pp. 103—109.
- [3] ÁDÁM, A., On some generalizations of cyclic networks, *Acta Cybernet.*, v. 1, 1971, pp. 105—119.
- [4] ÁDÁM, A., On graphs satisfying some conditions for cycles, I., *Acta Cybernet.*, v. 3, 1976, pp. 3—13.
- [5] ÁDÁM, A., On graphs satisfying some conditions for cycles, II., *Acta Cybernet.*, v. 3, 1977, pp. 69—78.
- [6] ÁDÁM, A., & J. BAGYINSZKI, On some enumeration questions concerning trees and tree-type networks, *Acta Cybernet.*, v. 1, 1972, pp. 129—145.
- [7] ÁDÁM, A., & U. KLING, On the behaviour of some cyclically symmetric networks, *Acta Cybernet.*, v. 1, 1971, pp. 69—79.
- [8] BERGE, C., *Graphes et hypergraphes*, Dunod, Paris, 1970.
- [8a] BERGE, C., *Graphs and hypergraphs*, North-Holland, Amsterdam, 1973.
- [9] ĐOKOVIĆ (DOKOVIĆ), D. Ž., Isomorphism problem for a special class of graphs, *Acta Math. Acad. Sci. Hungar.*, v. 21, 1970, pp. 267—270.
- [10] ELSPAS, B. & J. TURNER, Graphs with circulant adjacency matrices, *J. Combinatorial Theory*, v. 9, 1970, pp. 297—307.
- [11] ERDŐS, P. & A. RÉNYI, On the evolution of random graphs, *MTA Mat. Kutató Int. Közl.* (= *Publ. Math. Inst. Hung. Acad. Sci.*), v. 5, 1960, pp. 17—61.²²
- [12] HARARY, F., *Graph theory*, Addison-Wesley, Reading (Mass.), 1969.
- [12a] Харари, Ф., *Теория графов*, Мир, Москва, 1973.
- [13] KLING, U., Simulation neuronaler Impulsrhythmen (Zur Theorie der Netzwerke mit cyklischen Hemmverbindungen), *Kybernetik*, v. 9, 1971, pp. 123—139.
- [14] KLING, U. & G. SZÉKELY, Simulation of rhythmic nervous activities, I. (Function of networks with cyclic inhibitions), *Kybernetik*, v. 5, 1968, pp. 89—103.

²² Reprinted in the volumes:

ERDŐS, P., *The art of counting*, M. I. T. Press, Cambridge (Mass.), 1973, pp. 574—617,
RÉNYI, A., *Selected papers II.*, Akadémiai Kiadó, Budapest, 1976, pp. 482—525.

- [15] KONIKOWSKA, B., Continuous machines, *Information and Control*, v. 22, 1973, pp. 353—372.
- [16] TOIDA, S., A note on Ádám's conjecture, *J. Combinatorial Theory (Series B)*, in print.
- [17] ПОНТРИАГИН, Л. С., *Обыкновенные дифференциальные уравнения*, Физматгиз, Москва, 1961 (second edition: Наука, Москва, 1965).
- [17a] PONTRYAGIN, L. Sz., *Közönséges differenciálegyenletek*, Akadémiai Kiadó, Budapest, 1972.
- [17b] PONTRYAGIN, L. S., *Ordinary differential equations*, Addison-Wesley and Pergamon, Reading (Mass.) and London, 1962.
- [18] BABAI, L., Isomorphism problem for a class of point-symmetric structures, *Acta Math. Acad. Sci. Hungar.*, v. 29, 1977, pp. 329—336.

(Received June 10, 1976)

Topological analysis of linear systems

By I. PÁVÓ

Abstract



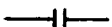


The author deals with the solution of the input—output analysis problem of the linear system models. Beside the traditional elements a suggestion is presented for introducing degenerate elements, hereby a more general class of practical linear systems can be taken into account. For the analysis by topological formulas the author gives k -trees generation procedures which are essentially applications of his earlier methods written in papers [3] and [4]. Finally concrete examples are presented from the area of the electrical networks model as well.

Introduction

In this paper we are going to deal with models of linear systems which can be described by differential equations in general, and can disjoin two terminal tools connecting two points or domains. In practice we can find such systems at electrical, mechanical, pneumatic, thermodynamic, etc. networks. After defining the system elements let the structure of the linear system be given by an abstract graph. Beside the conservative system elements well-known from reference [9], degenerate elements will be introduced by which the model of more general linear systems can be given. For example, it can be shown that any two terminal or two part electrical network consisting of passive elements, mutual inductances and controlled generators may be modelled as a network consisting only of degenerate (nullator and norator) and passive elements [8]. The network determinant is set in the centre of the formal solution of the system equations, a suggestion will be presented for its calculation by a topological formula with application of the generation of k -trees suggested by the author in earlier papers [3] and [4]. Finally, the order of the topological analysis will be shown by concrete examples for the calculations of the electrical linear systems.

The elements of the linear system

In the model of the linear system variables of two type are allowed; namely, *through* and *across*, which are characterized by the usual measuring directions [9]. (In practice through variables are electrical current, power, flood, convection of heat, etc., across variables are voltage, rotation, pressure, temperature, etc.). In general a variable is a function of the time, and we assume that there exists its Laplace-transform. The system elements are defined by relations between the through and the across variables in the following manner:

Type of the relation.	Name of the element	Sign of the element	Relation between through and across variables
Inductive	Inductance		L. s. $J(s) = V(s)$
Resistive	Resistance		R. $J(s) = V(s)$
Capacitive	Capaticy		$J(s) = C. s. V(s)$
Over determined	Nullator		$J(s) = 0, V(s) = 0$
Undetermined	Norator		$J(s) = \text{arbitrary}, V(s) = \text{arbitrary}$

The symbol s is the complex value, J and V are the Laplace-transforms of the corresponding functions (through and across variables). The equations between the Laplace-transforms are valid under zero initial conditions. L , R and C are arbitrary real numbers differing from zero (they are the parameters of the corresponding system elements). In case of a concrete system $V(s)$ and $J(s)$ belonging to a norator are defined by the other elements of the system, the word "arbitrary" is to be understood in this manner.

The first three system elements are to be regarded as classical ones from the reference. These are the so called "passive elements". Now we give a reason for the introduction of degenerate elements (nullator and norator).

The through and across source variables driving the linear systems (the independent generators) are given by their functions. Beside such ideal source variables other ones may occur as well, the function of which depend on through or across variable between two vertices of the system graph (controlled generators). For example, it may occur that the controlled through source variable between vertices i and k is the multiple of the across variable between vertices l and k . This is the situation with electrical linear networks in case of voltage controlled circuit generators. The conventional sign of the controlled source variable occurring in the present example and its nullator—norator pair equivalent network are shown by Fig. 1. According to Fig. 1 such an equivalent network may be produced from an unique passive element (resistance) and from a nullator—norator pair.

After the introduction of the nullator—norator pairs there is a possibility for producing the models of the controlled generators of all types and the ideal transformer by electrical networks [7]. In the nullator—norator pair model of a general linear network only elements with parameters R , L and C , nullators and

norators, (the letters as a pair) occur [6]. Thus by the introduction of degenerated element suggested in the present paper, a practically larger class of the linear systems may be described than by the set of passive elements.

We do not define exactly the rules of the connections between the system elements. But we assume in the present paper that the graph of the system is connected,

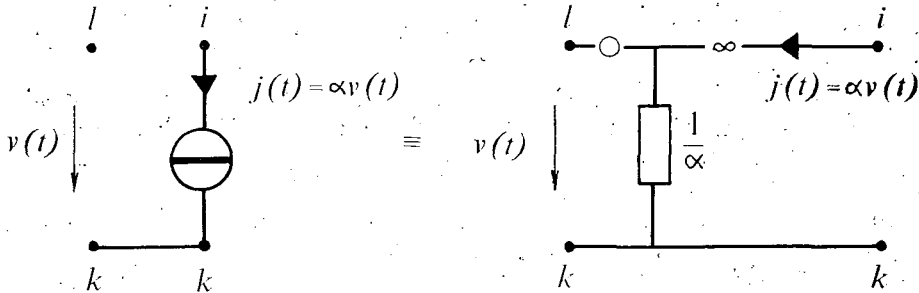


Fig. 1

an element does not contain a loop, the degenerated elements occur only in pairs and degenerated element cannot be parallel with any passive one. According to [8] the latter condition does not break the general case. Practically, concrete linear systems obviously hold these conditions.

After this we draw up the program of the input—output analysis of linear systems in the following manner. Consider a model of a linear system by its graph, the edges of which are system elements, through and across source variables driving the system. Determine the concrete values of the through and across variables in each passive elements.

System equations and their formal solution

Let the system graph consist of n vertices, l edges containing a passive element and N pairs of nullator—norator edges. The equivalent network of an edge of the graph containing a passive element is shown in Fig. 2 (Fig. 2 takes into account the generalized case).

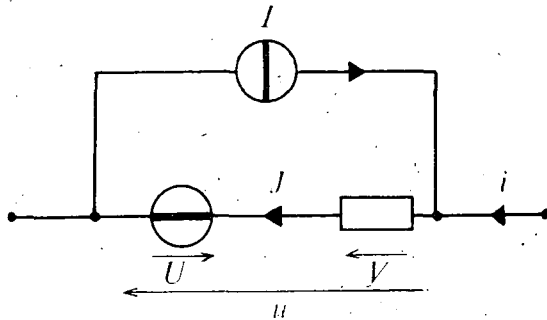


Fig. 2

Concerning the passive element the vector equations are the following:

$$\mathbf{V} = \mathbf{u} + \mathbf{U} \quad (1)$$

$$\mathbf{J} = \mathbf{i} + \mathbf{I}, \quad (2)$$

where \mathbf{U} and \mathbf{I} are vectors of the across and through source variables, respectively, the components of which can be found in the equivalent networks of the edges of the graph, \mathbf{u} and \mathbf{i} are column vectors of size l made from the Laplace-transforms of the across and through variables of the edges.

After suitable numbering of the edges, let \mathbf{A} be the incidence matrix of the graph concerning a reference vertex, and divide it into the following parts:

$$\mathbf{A} = [\mathbf{A}_p \ \mathbf{A}_0 \ \mathbf{A}_\infty]$$

where submatrix \mathbf{A}_p corresponds to the edges containing a passive element, \mathbf{A}_0 to the nullator and \mathbf{A}_∞ to the norator edges.

It is true [cf. 8] that

$$\mathbf{J} = \mathbf{yV} \quad (3)$$

where $\mathbf{y} = \langle y_1, y_2, \dots, y_l \rangle$ is a diagonal matrix and y_i is the operator admittance of the i -th passive element. Namely,

$$y_i = \frac{J_i(s)}{V_i(s)}$$

After this we define the incidence matrices of the modified graphs.

Let ${}_0\mathbf{A}_p$ represent the reduced incidence matrix of the graph which is determined by the edges containing a passive element after short circuiting all nullator edge endpoints. We use matrix ${}_\infty\mathbf{A}_p$ in a similar sense, i.e., let ${}_\infty\mathbf{A}_p$ be the reduced incidence matrix of the graph which is determined by the edges containing a passive element after short circuiting all norator edges endpoints.

To describe the linear system we write the law of the node [8] in the following form:

$${}_\infty\mathbf{A}_p \cdot \mathbf{i} = \mathbf{0}. \quad (4)$$

Finally, let us introduce the vector \mathbf{P} of size $(n-N)$ by the equation

$$\mathbf{u} = {}_0\mathbf{A}_p^t \cdot \mathbf{P} \quad (5)$$

the components of which are sum of the across variables along the path connecting the suitable vertex of the graph with the reference vertex [9].

(1)–(5) are the basic equations of the examined system.

Considering (1) and (2),

$$\mathbf{i} + \mathbf{I} = \mathbf{y}(\mathbf{u} + \mathbf{U}). \quad (6)$$

Let us multiply (6) by the matrix ${}_\infty\mathbf{A}_p$ from the left, and consider (4). Then

$${}_\infty\mathbf{A}_p \cdot \mathbf{I} = {}_\infty\mathbf{A}_p \cdot \mathbf{y} \cdot \mathbf{u} + {}_\infty\mathbf{A}_p \cdot \mathbf{y} \cdot \mathbf{U}. \quad (7)$$

Taking into account (5), after some rearrangement we get

$${}_\infty\mathbf{A}_p \cdot \mathbf{y} \cdot {}_0\mathbf{A}_p^t \cdot \mathbf{P} = {}_\infty\mathbf{A}_p (\mathbf{I} - \mathbf{y} \cdot \mathbf{U}). \quad (8)$$

Let us introduce the symbol \mathbf{Y} by

$$\mathbf{Y} = {}_{\infty}\mathbf{A}_p \cdot \mathbf{y} \cdot {}_0\mathbf{A}_p^t.$$

If $\det(\mathbf{Y}) \neq 0$ then there exists the inverse matrix \mathbf{Y}^{-1} . In this case, by [8], we can write

$$\mathbf{P} = \mathbf{Y}^{-1} \cdot {}_{\infty}\mathbf{A}_p \cdot (\mathbf{I} - \mathbf{y} \cdot \mathbf{U}). \quad (9)$$

One can consider (9) as the explicit solution of the input—output analysis. We remark that in this manner the calculation of $\det(\mathbf{Y})$ is necessary for the solution, or rather, for determining the inverse of a matrix.

If the system contains also purely across or through source variable edges then the form of (9) is modified slightly. There is no problem that the graph contains only through variable edges. Namely, in this case the equivalent network of the generalized passive edge has a passive element with "operator admittance of zero". It can be seen that in the case of across source variable edges the right side of (9) is invariable, and the modification of (9) does not influence the calculation of $\det(\mathbf{Y})$. Further more, $\det(\mathbf{Y})$ will be called *system determinant*.

Calculation of the system determinant

It is known from [5] that

$$\det(\mathbf{Y}) = \sum F^{N+1}. \quad (10)$$

This is a topological formula, where F^{N+1} is an edge admittance product formed by an $(N+1)$ -tree of the system graph which consists of only edges containing a passive element, and this $(N+1)$ -tree turns into a circuitless connected graph after short-circuiting either the nullator or the norator endpoints. The sign of each product comes from the product of the corresponding minors of ${}_{\infty}\mathbf{A}_p$ and ${}_0\mathbf{A}_p^t$. The summation takes into account the same $(N+1)$ -trees of the type in question. The generation of such $(N+1)$ -trees can be given by the following algorithm.

Step 1. After missing all nullator edges produce and list trees of the remained graph which contain all the norator edges. This is possible by the suitable organization of the method written in [3].

Step 2. Leave the norator edges from the trees produced by step 1. We obtain $\{F^{N+1}(N+1)\}$ -trees, and

$$\{F^{N+1}\} \cong \{F^{N+1}\} \quad (11)$$

is obviously fulfilled.

Step 3. Short-circuit the endpoints of the left nullator edges in each of the elements of set $\{F^{N+1}\}$. Let us select from them the circuitless graphs according to method [4]. By this the generation of all F^{N+1} comes to the end.

Finally, we remark that the sign of any F^{N+1} edge admittance product can be determined by the application of the Davies rule.

Universal parameters

We use the method written in this paper in the analysis of an electrical linear network model consisting of nullator—norator pairs. It is known that any universal parameter of a two port network (if there exists any) can be obtained as system determinant of the two port network its input and output being closed by suitable nullator—norator pairs [2]. One can see the suitable closure in Fig. 3 together with

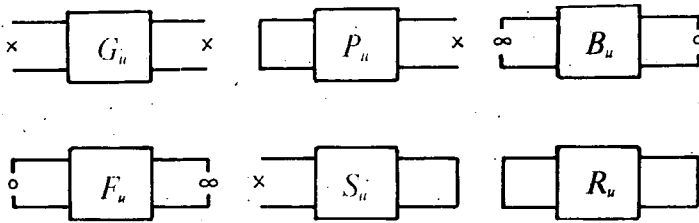


Fig. 3

the symbol of the universal parameter. Notice that the short circuit closure in Fig. 3 is equivalent to a parallel connected nullator—norator pair. Referring to the earlier, it is clear that for the production of any universal parameter by topological formulas we need the earlier k -trees of the suitable closed network graph, and now $N+1 \cong \cong k \cong N+3$, where N is the number of the nullator—norator pairs in the original network model.

We give a block scheme of the k -trees generation in Fig. 4 to produce an arbitrary universal parameter. To realize this algorithm by a computer the procedure has all the advantages of methods written in [3] and [4] (i.e., “calculation of the trees” is possible “one by one”, it is not necessary to reserve them in the storage capacity).

In the case of active networks the determination of the universal parameters makes possible to describe the system functions of the network model in question, which are quotients of the suitable universal parameters in general [2].

Further on we are going to study the description of some system functions either passive or active networks.

Analysis of passive networks

First consider a two terminal network consisting of R , L and C elements and set ourselves an aim to write its operator admittance function in the general case. The task can be drawn up as the determination of the input admittance of a two terminal network closing its output by break (the output may be an arbitrarily chosen pair of the vertex in the network).

The first equation of the inverse hybrid characteristics is:

$$I_1 = D_{11}U_1 + D_{12}I_2$$

from which after considering $I_2=0$ we get

$$Y_{in} = \frac{I_1}{U_1} = D_{11}.$$

Taking into account the definitions of the universal parameters we can write

$$Y_{in} = \frac{G_u}{P_u}. \quad (12)$$

Taking into account the calculation of G_u and P_u by topological formulas (see Fig. 3) we can see that to produce the numerator of (12) all the trees of the network graph are needed, while the denominator needs all 2-trees which contain the input points in separate components of the graph. So k -trees needed to (12) can be generated by the somewhat modified method written in [3]. Next determine the transfer impedance function of an arbitrary RLC two port network, the scheme of which is represented in Fig. 5.

Now from the impedance characteristics of the network closed on the output by a break we obtain

$$Z_{tr} = \left. \frac{U_2}{I_1} \right|_{I_2=0} = Z_{21}, \quad (13)$$

and from (13), because of the definitions of the universal parameters,

$$Z_{tr} = \frac{B_u}{G_u} \quad (14)$$

follows.

To write the numerator of (14) all the 2-trees are needed, which separate either the input or the output vertices (i.e., the 2-trees contain these points in different components). From the latter statement it follows that to determine B_u the closed

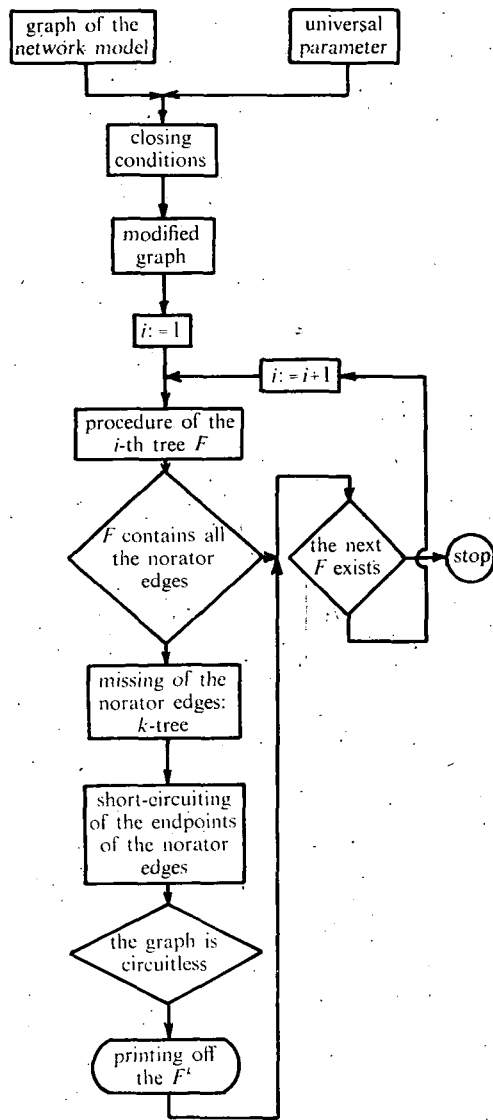


Fig. 4

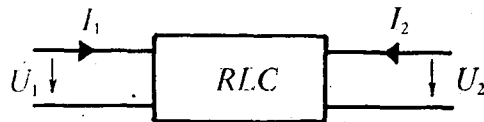


Fig. 5

network model contains a unique nullator—norator pair each degenerate element of which is connected to the input and output. It is not a problem to generate k -trees for the denominator of (14) (see the first example). We remark that similar k -trees are needed for producing F_u as in the case of B_u .

Network containing controlled generator

As a concrete example let us consider the two port network in Fig. 6 consisting of an ideal operational amplifier, and set as a task to generate k -trees necessary

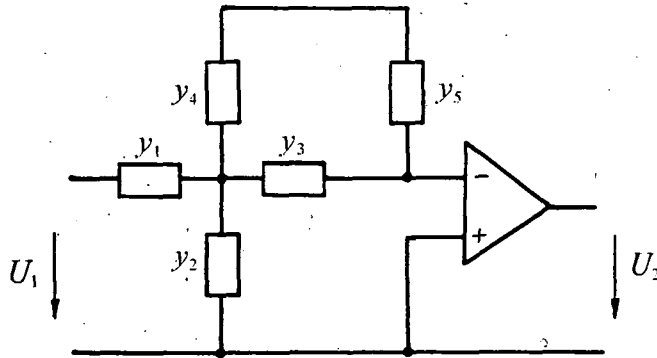


Fig. 6

to the calculation of the transfer voltage function of the network by topological formulas from the suitable network model.

From the inverse hybrid characteristics of a two port network whose output is closed by a break it follows:

$$A_u = \left. \frac{U_2}{U_1} \right|_{I_2=0} = D_{21}, \quad (15)$$

Taking into account the definitions of the universal parameters, from (15) we get

$$A_u = \frac{B_u}{P_u}. \quad (16)$$

After using the nullator—norator equivalent network of the ideal operational amplifier we can see the network model in Fig. 7. To determine the denominator of (16) close the input of the network model by a parallel connected nullator—norator pair. Thus we get the task discussed in paper [4], and the results are 3-trees

(01400), (03400), (04200), (04400) and (05400)

in order.

To determine the numerator of (16) close the network model shown in Fig. 7 by a norator on the input and by a nullator on the output. The modified network

model is shown by Fig. 8. It can be seen, if we search the trees of the modified graph containing all the norator edges and left the norator edges from them, that the obtained 3-trees are equivalent to the 3-trees before the short circuiting listed in paper [4]:

(01200), (01400), (03400), (04200), (04400), (05200), (05400).

But at present the condition of the short circuiting differs from the earlier one. Namely,

$$a = 3 = 4 = 5$$

in respect to the unique symbol element.

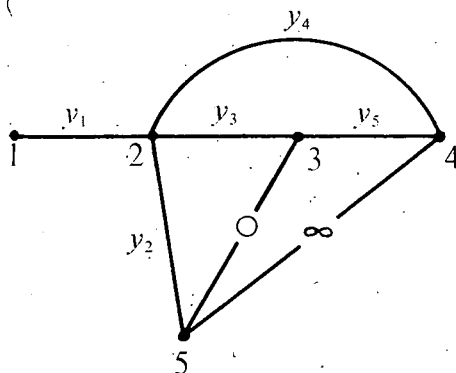


Fig. 7

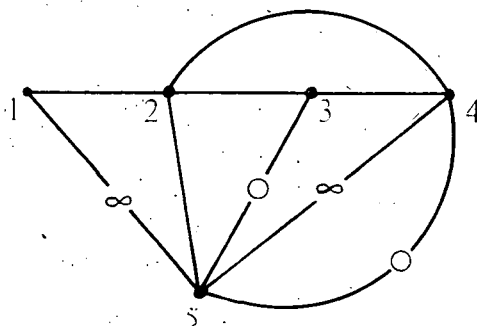


Fig. 8

To decide which subgraphs are circuitless write in a table the row vector representations. The first common row of the representations is

$$1 \quad 2 \quad a \quad a \quad a,$$

while the second rows are in order

0	1	2	0	0
0	1	a	0	0
0	a	a	0	0
0	a	2	0	0
0	a	a	0	0
0	a	2	0	0
0	a	a	0	0

Performing their complete cycle check, only the first representation leads to a finite outcome and the suitable reduced graph is circuitless as well.

We get that the numerator of (16) needs only one 3-tree with representation (01200). According to topological formula (16), taking into account the sign of

the edge admittance products, after some calculation we have got for the active network

$$\frac{U_2}{U_1} = - \frac{y_1 y_3}{y_5 (y_1 + y_2 + y_3 + y_4) + y_3 y_5}$$

RESEARCH GROUP ON MATHEMATICAL LOGIC
AND THEORY OF AUTOMATA OF THE
HUNGARIAN ACADEMY OF SCIENCES
H-6720 SZEGED, HUNGARY
SOMOGYI U. 7.

References

- [1] DAVIES, A. C., Matrix analysis of networks containing nullators and norators, *Electronics Letters*, 1966, No. 2, p. 48, No. 3, p. 91.
- [2] HENNYEI, Z., *Lineáris áramkörök elmélete* (Theory of the linear circuits), Akadémiai Kiadó, Budapest, 1958.
- [3] PÁVÓ, I., Generation of the k -trees of a graph, *Acta Cybernet.*, Szeged, v. 1, 1971, pp. 57—68.
- [4] PÁVÓ, I., Short-circuited k -trees, *Acta Cybernet.*, Szeged, v. 2, 1975, pp. 323—333.
- [5] SZEPESI, T. & M. GUTTERMUTH, Lineáris aktív hálózatok topológiai analízise (Topological analysis of linear active networks), *Híradástechnika*, Budapest, v. 13, 1972, pp. 56—61.
- [6] VÁGÓ, I., Calculation networks models containing nullators and norators, *Periodica Polytechnica*, Budapest, v. 17, 1973, pp. 311—319.
- [7] VÁGÓ, I. & E. HOLLÓS, Two-port models with nullators and norators, *Periodica Polytechnica*, Budapest, v. 17, 1973, pp. 300—309.
- [8] VÁGÓ, I., *A gráfelmélet alkalmazása villamos hálózatok számításában* (The application of the graphtheory in the calculation of the electrical networks), Műszaki Könyvkiadó, Budapest, 1976.
- [9] ZADEH, L. A. & E. POLAK, *System theory*, McGraw-Hill Co., Inter-University Electronics Series, v. 8, 1970.

(Received May 11, 1977)

On some basic theoretical problems of fuzzy mathematics

By L. T. KÓCZY

Abstract

The aim of this paper is to raise some questions — and partly, also to answer them — in connection with two important problem groups of fuzzy mathematics: n -fuzzy objects and the sigma-properties of different interactive fuzzy structures. These questions are suggested by the analysis of natural languages, the common sense thinking — which are typical fields where the most adequate mathematical model is a fuzzy one —, especially by complex adjectival structures and subjective “verifying” processes, respectively. They have, however, a real practical significance also in the field of engineering, as e.g. in learning machine problems.

In the first part we try to point to the practical importance of the concept of fuzzy objects of type n (or n -fuzzy objects), from the aspect of modelling natural languages. A useful way to define n -fuzzy algebras, i.e., generalizing ordinary fuzzy algebras for n -fuzzy objects, is also given, with introducing an isomorphism mapping from the fuzzy object space to the n -fuzzy object space. As an example, an R - n -fuzzy algebra is defined. Because of the isomorphic property of the above mapping the later studies can be restricted to ordinary fuzzy objects.

In the second part some very basic concepts in connection with the sigma-properties of fuzzy algebras are given and some simple theorems are proved. These are quite important from the aspect of fuzzy learning processes, as their probability theoretic interpretation leads to several convergence theorems — which are not dealt with here, however.

In this part we introduce the concept of the quantified algebra of a fuzzy algebra, and by means of this concept a close relation between interactive fuzzy and Boolean algebras is proved different from the relation between non-interactive system and Boolean algebra given by Zadeh.

Although any presentation of complete application examples is not at all intended in this paper, some aspects of the application of the above results, especially in learning control algorithms, are given, the statements backed up by the experience of a simulation experiment going on at present.

1. Introduction

In the invention of fuzzy mathematics, one of the most important aspects was the intention of obtaining an effective way for modelling badly defined phenomena appearing over and over in the everyday life and many fields of sciences. Modelling by traditional methods proved to be often very coarse. The first problems of this type were raised in pattern recognition by L. A. Zadeh [1, 2]. However, this field turned out soon not to be the one where the application of his new concept made the quickest advance. In disciplines having an even more human factor as linguistics, economics, etc. results could be produced easier. Nevertheless, it did not mean that fuzzy concepts played no part in engineering; rather, that engineering had had its own well-worked-up mathematical background, and to replace it — if only partly — by a new model necessarily met resistance and obstacles. We have to admit it, too, that the bases of fuzzy mathematics have been laid often inexactly which fact resulted, as a matter of course, in the increase of resistance.

As many other, we were suggested by such failures of exactness to look for a correct formulation of fuzziness, by finding and observing real objects and phenomena corresponding to the basic concept of fuzzy objects and operations on them. For this purpose, some phenomena in the common sense thinking and the natural languages turned out to be very suitable. On the basis of these considerations we established a group of fuzzy algebras, among them the most important one was R -fuzzy algebra [3, 4].

Some inference methods used in the medical science gave the basic aspects in our constructing these systems; they also pointed to the fact, that the fuzzy operations used formerly (max—min or non-interactive ones) were in accordance only with a restricted part of fuzzy objects. The new type of operations turned out to be much more adequate and applicable as it had been proved by some simple experiments in cluster analysis and learning control carried out by the author and his colleagues.

We now introduce some notions and notations.

I. Pre- R -fuzzy algebra

1. a) There exists a nonempty set

$$X = \{x\},$$

which is named *universe* or *base set*.

- b) There exists a nonempty set

$$\otimes = \{A\}$$

the elements of which are named *fuzzy objects*.

- c) There exists a mapping M , so that

$$M: \otimes \rightarrow \mathcal{F},$$

where $\mathcal{F} = \{\mu | \mu: X \rightarrow \mathcal{P}\}$,

where \mathcal{P} is an ordered nonempty set.

(In the practice, the usual representation of \mathcal{P} is the closed interval $[0, 1]$,

then the functions μ and the single values $\mu(x)$ are named membership functions and membership grades, respectively.)

Definition. We define $A=B$ (A equal to B) such that it is the abbreviation for

$$M(A) = M(B).$$

(In the practice it means that

$$\mu_A(x) = \mu_B(x),$$

for all $x \in X$.)

2. a) There exist A and B in \otimes such that $A \neq B$,
- b) \otimes is closed under the binary operation \vee , named *disjunction*,
- c) \otimes is closed under the binary operation \wedge , named *conjunction*,
- d) \otimes is closed under the unary operation \neg , named *negation*.
3. a) $A \vee B = B \vee A$, for all $A, B \in \otimes$.
- b) $(A \vee B) \vee C = A \vee (B \vee C)$, for all $A, B, C \in \otimes$.
- c) $\neg \neg A = A$, for all $A \in \otimes$.
- d) There exists an element $\emptyset \in \otimes$, named zero, such that for all $A \in \otimes$,

$$d1) \quad A \vee \emptyset = A,$$

$$d2) \quad A \wedge \emptyset = \emptyset.$$

- e) $\neg(A \vee B) = \neg A \wedge \neg B$, for all $A, B \in \otimes$.
4. a) $M((A \wedge B) \vee (A \wedge C)) > M(A \wedge (B \vee C))$, for all $A, B, C \in \otimes$, if

$$(A \wedge B) \vee (A \wedge C) \neq \emptyset \quad \text{and} \quad A \wedge (B \vee C) \neq \neg \emptyset.$$
- b) $M((A \vee B) \wedge (A \vee C)) < M(A \vee (B \wedge C))$, for all $A, B, C \in \otimes$, if

$$(A \vee B) \wedge (A \vee C) \neq \neg \emptyset \quad \text{and} \quad A \vee (B \wedge C) \neq \emptyset.$$
- c) $M(A \vee B) > M(A)$, for all $A, B \in \otimes$, if $A \neq \neg \emptyset$ and $B \neq \emptyset$.
- d) $M(A \wedge B) < M(A)$, for all $A, B \in \otimes$, if $A \neq \emptyset$ and $B \neq \neg \emptyset$.
- e) $M(A) - M(B) = M(\neg B) - M(\neg A)$, for all $A, B \in \otimes$.

5. a) If the equation $A \vee U = B$ ($A, B \in \otimes$, $A \neq \neg \emptyset$) has a solution $U \in \otimes$, than U is unique.
- b) If the equation $A \wedge U = B$ ($A, B \in \otimes$, $A \neq \emptyset$) has a solution $U \in \otimes$, than U is unique.
6. a) $M(A \vee B)$ is a continuously differentiable function in terms of $M(A)$ and $M(B)$.
- b) $M(A \wedge B)$ is a continuously differentiable function in terms of $M(A)$ and $M(B)$.
- c) $M(\neg A)$ is a continuously differentiable function in terms of $M(A)$.

II. R-fuzzy algebra

1. Like in System I. (Here the usual representation of \mathcal{P} is R^1 .)
- 2—4). Like in System I.
5. a) Equation $A \vee U = B$ ($A, B \in \otimes$, $A \neq \neg \emptyset$) can be solved for $U \in \otimes$ and U is unique.

- b) Equation $A \wedge U = B$ ($A, B \in \otimes, A \neq \emptyset$) can be solved for $U \in \otimes$ and U is unique.
6. Like in System I.

III. S-fuzzy algebra

- 1—3. Like in System I.
4. a) $M(A \vee A) > M(A)$, for all $A \in \otimes, A \neq \emptyset, A \neq \neg \emptyset$.
 b) $M(A \wedge A) < M(A)$, for all $A \in \otimes, A \neq \emptyset, A \neq \neg \emptyset$.
 c) $M(A \vee A) > M(B \vee B)$ iff $M(A) > M(B)$; $A, B \in \otimes$.
 d) $M(A \wedge A) > M(B \wedge B)$ iff $M(A) > M(B)$; $A, B \in \otimes$.
5. Like condition 6 in System I.

At the end of the presentation of our axiomatic systems we should like to stress the fact, that all three systems are symmetrical for the operations conjunction and disjunction, that is, they are dual structures — in spite of their asymmetrical appearance.

Although the above systems were constructed merely on the basis of theoretic speculations and passive observations of the rules of common sense inference, after they having been established in a more or less exact form, they would be backed up by some independent, objective, experimental facts discovered by another author [5 and 6, respectively]. Also some independent theoretical considerations pointed to the fact that adequate fuzzy axiomatics must be constructed similarly to I. (Although no dual systems were introduced.) [7]. On the basis of these results we were able to extend the group of representative operation trebles to a very general class (rational functions) [8]. As we are not concerned now with the problem of representatives, we omit a detailed presentation of it.

2. N-fuzzy objects

When fuzzy sets, fuzzy objects were introduced by L. A. Zadeh, they seemed general enough for modelling “all fuzzy-type phenomena of the world” [2]. Later analyses of natural languages etc., however, resulted in the perception of the fact, that a more general concept — modelling twofold, n -fold fuzziness — must be introduced: this has been done also by Zadeh, and has been named “fuzzy object of type n ” or more simply “ n -fuzzy object” [9]. Although this first formulation was not at all exact, the general idea can be preserved, however, formulating its definition in an entirely different way. As we see it now, this concept is the most general one (having also a practical importance), so we are very much interested in it. In this section we shall present a way, how handling of n -fuzzy objects can be simplified considerably, restricting the field of interest again to ordinary fuzzy systems (as I, II, and III).

According to the axiomatics an ordinary fuzzy object can be given in the following way:

Let X be the given universe, then A_X is as follows:

$$\langle X, qA(x) \rangle,$$

where

$$qA: X \rightarrow \mathcal{P} \quad (\mathcal{P} \text{ is } [0, 1] \text{ in the most simple case.})$$

Let us consider now an example from the natural language:

“Peter is tall.”

and

“Peter is rather tall.”

are both fuzzy statements. We are interested now in the predicative part of them. If we define X as

$$X = \{\text{possible values of the height of a man}\},$$

“tall” is a fuzzy set of X (A_X). What is “rather”? It is also a fuzzy expression, but a fuzzy set of *grades*. Let now P be the universe of abstract truth grades mapped on the interval $[0, 1]$. Then “rather” is a fuzzy set B_P , a double consisting of P and a function $q_B(p)$. We must become aware of the fact, that taking a singular x in X , qA maps it on the ordered set of truth values \mathcal{P} , which is identical to P . Thus if we consider an object gained by “modulating” $qA(x)$ by $qB(p)$, we get a mapping ($qA^2(x)$) which maps X on the set of fuzzy subsets of \mathcal{P} . This is named 2-fuzzy membership function of the expression “rather tall”. A 2-fuzzy object is now the system

$$A_X^2 = \langle X, qA^2(x) \rangle.$$

After this example we can give the exact definition of an n -fuzzy object:

Definition. Given a universe $X = \{x\}$. Then $S_X^n = \langle X, qS^n \rangle$, is an n -fuzzy object of X where

$$\begin{aligned} qS^n: X &\rightarrow S_{\mathcal{P}}^{n-1}, \\ S_{\mathcal{P}}^{n-1} &= \langle \mathcal{P}, qS^{n-1} \rangle, \\ qS^{n-1}: \mathcal{P} &\rightarrow S^{n-2}, \\ &\vdots \\ S_{\mathcal{P}}^1 &= S_{\mathcal{P}} = \langle \mathcal{P}, qS \rangle, \\ qS: \mathcal{P} &\rightarrow \mathcal{P}. \end{aligned}$$

A much greater problem is the proper way of defining operations over n -fuzzy objects. It has been done by Zadeh, we must state, however, that his definition is not correct, as the set of n -fuzzy objects is not closed under his disjunction (and conjunction either.) Another way has been taken by Mizumoto *et al.* [10], where no failure of correctness appeared, nevertheless, the properties of n -fuzzy operations contradicted the simplest natural properties of normal operations. We mention only the fact, that in Mizumoto’s disjunction, e.g., it is possible that the membership function of the result (taken over one single x) is *less than both* functions of the arguments.

Here, we are going to present a way for defining an arbitrary operation over n -fuzzy objects, which has been introduced over fuzzy objects, in such a way that this definition is always consistent and in accordance with the original one.

Definition. Let $*$ be an arbitrary m -ary operation over fuzzy objects such that

$$\begin{aligned} * (A_1, A_2, \dots, A_m) &= * (\langle X, qA_1 \rangle, \langle X, qA_2 \rangle, \dots, \langle X, qA_m \rangle) = \\ &= \langle X, F(qA_1, qA_2, \dots, qA_m) \rangle, \end{aligned}$$

$$\text{where } F: M^m \rightarrow M, \quad M = \{q\},$$

$$\text{and } F(qA_1, qA_2, \dots, qA_m)x = f(qA_1(x), qA_2(x), \dots, qA_m(x)),$$

$$\text{where } f: \mathcal{P}^m \rightarrow \mathcal{P}.$$

Let now C_X^n be an n -fuzzy object of X such that

$$C_X^n = \langle X, qC^n \rangle,$$

where

$$qC^n(x) = \langle P_1, qC_x^{n-1} \rangle,$$

$$qC_x^{n-1}(p_1) = \langle P_2, qC_{x, p_1}^{n-2} \rangle, \quad p_1 \in P_1,$$

$$\vdots$$

$$qC_{x, p_1, p_2, \dots, p_{(n-2)}}(p_{(n-1)}) = qC(x, p_1, p_2, \dots, p_{(n-1)});$$

$$P_1 = P_2 = \dots = P_{n-1} = \mathcal{P}.$$

Let C be the fuzzy "equivalent" of C_X^n :

$$C = \langle Xx\mathcal{P}^{n-1}, qC \rangle = \langle Y, qC \rangle, \quad \text{and } Y = \{y\} = \{(x, p_1, \dots, p_m)\}.$$

Now, $*$ is to be extended in the following way:

Let E be a one-to-one mapping from the set of n -fuzzy objects of X to the set of fuzzy objects of Y such that

$$E(C_X^n) = C.$$

Then

$$\begin{aligned} * (A_{1X}^n, A_{2X}^n, \dots, A_{mX}^n) &= * (\langle X, qA_1^n \rangle, \langle X, qA_2^n \rangle, \dots, \langle X, qA_m^n \rangle) = \\ &= \langle X, q * (A_1, A_2, \dots, A_m)^n \rangle \end{aligned}$$

means, that

$$q * (A_1, A_2, \dots, A_m)^n(y) = f(qE(A_{1X}^n)(y), qE(A_{2X}^n)(y), \dots, qE(A_{mX}^n)(y)).$$

This is equivalent to

$$* (A_{1X}^n, A_{2X}^n, \dots, A_{mX}^n) = E^{-1}(* (E(A_{2X}^n), E(A_{2X}^n), \dots, E(A_{mX}^n))).$$

Although the above definition might seem to be quite complicated at first sight, in reality it is very simple: the operations must be computed point by point in n -fuzzy cases.

On the basis of the above formulae, a very important fact can be proved in a most simple way:

Theorem (of isomorphism). Let \otimes^n be an algebra with r operations of the n -fuzzy objects of X . Moreover, let these r operations be extensions of r similar operations of the algebra \otimes of the fuzzy objects of $Xx^{\mathcal{P}^{n-1}}=Y$. If

$$E(S_X^n) = S_Y \text{ for all } S_X^n \in \otimes^n, S_Y \in \otimes \text{ then}$$

\otimes and \otimes^n are isomorphic.

We close this part with a simple example which shows that, using this theorem, the computation in the field of n -fuzzy objects becomes quite simple, proving also that no separate examinations or considerations are necessary when dealing with n -fuzzy problems: All results concerning ordinary fuzzy algebras are also valid for their n -fuzzy equivalents. We shall utilize this fact in the second part.

Example. Let us define disjunction over fuzzy objects as follows:

$$qA \vee B = qA + qB - qA \cdot qB.$$

We represent the interval $[0, 1]$ by 6 points only: 0.0, 0.2, 0.4, 0.6, 0.8, 1.0.

$$X = \{a, b, c, d\}.$$

In the table we give the membership functions of two objects of $X: A_X^2$ and B_X^2 .

	P	0.0	0.2	0.4	0.6	0.8	1.0	
$A_X^2:$	X							
	a	0.0	0.0	0.1	0.3	0.7	0.9	$qA^2(a)$
	b	0.0	0.1	0.2	0.6	1.0	0.6	$qA^2(b)$
	c	0.0	0.2	0.6	1.0	0.9	0.4	$qA^2(c)$
	d	0.2	0.7	1.0	0.9	0.3	0.0	$qA^2(d)$

	P	0.0	0.2	0.4	0.6	0.8	1.0	
$B_X^2:$	X							
	a	0.0	0.3	0.9	1.0	0.7	0.5	$qB^2(a)$
	b	0.2	0.5	0.9	0.9	0.8	0.7	$qB^2(b)$
	c	0.4	0.7	0.9	0.9	0.9	0.9	$qB^2(c)$
	d	0.6	0.9	0.9	0.9	0.9	1.0	$qB^2(d)$

Using the given definition of the disjunction, $C_X^2 = A_X^2 \vee B_X^2$ can be given in the following way

	P	0.0	0.2	0.4	0.6	0.8	1.0	
$C_X^2:$	X							
	a	0.0	0.3	0.91	1.0	0.91	0.95	$qC^2(a)$
	b	0.2	0.55	0.92	0.96	1.0	0.88	$qC^2(b)$
	c	0.4	0.76	0.96	1.0	0.99	0.94	$qC^2(c)$
	d	0.68	0.97	1.0	0.99	0.93	1.0	$qC^2(d)$

3. Basic concepts of fuzzy sigma-algebras

In this section we introduce some very basic ideas in connection with fuzzy sigma-algebras and related problems. Then, we are able to deal with an interesting transformation of algebras named "quantification" which leads to the concept of a special type of convergent algebras.

First we give some definitions.

Definition. G is a fuzzy sigma-algebra, if G is a fuzzy algebra (in either senses of the Introduction, or in the sense, as it has been defined by Zadeh, etc.) and the infinite conjunction and disjunction are defined in G , i.e.,

$$\bigvee_{i \in H} A_i \in G, \quad \text{and} \quad \bigwedge_{i \in H} A_i \in G, \quad \text{where} \quad |H| = \aleph_0, \quad A_i \in G \forall i.$$

Thus, e.g., III and the sigma-properties together form an S -sigma-algebra.

Definition. Let A_i be a sequence of fuzzy objects

$$A_i = \langle X, q_i \rangle \quad (A_i \in G).$$

Then the limit value of A_i is $A = \langle X, q \rangle$ ($A \in G$),

if

$$q = \lim_{i \rightarrow \infty} q_i$$

in the ordinary sense. Thus A_i is convergent iff q_i is convergent.

Using this notion, the infinite conjunction or disjunction can be computed:

$$\bigvee_{i=1}^{\infty} A_i = \lim_{j \rightarrow \infty} B_j, \quad \text{where} \quad B_j = \bigvee_{i=1}^j A_i.$$

Definition. \exists_H is named the existential quantifier over the well-ordered set of indices H

$$\exists_H A_i = \bigvee_{i \in H} A_i \quad (A_i \in G).$$

Similarly, the universal quantifier over H , \forall_H is defined as

$$\forall_H A_i = \bigwedge_{i \in H} A_i \quad (A_i \in G).$$

Although this way of generalizing the notion of quantifiers of predicate calculus is not the only (and not at all the most logical) one. However, we shall need this special concept in our further examinations. Since it has a vague resemblance to quantifiers, thus it will not cause any ambiguity.

We mention also, that we will not deal with general case since we are interested only in the special case where

$$A_i = A_j, \quad \text{for all } i, j.$$

Definition. Let Q be the algebra defined in the following way: The elements of Q are gained from a fuzzy algebra G

$$Q' = \{\exists A | A \in G\} \cup \{\forall A | A \in G\},$$

where $\exists A$ and $\forall A$ are used for $\exists_H A_i (A_i = A, \forall i)$ and $\forall_H A_i (A_i = A, \forall i)$, respectively.

Since G is a fuzzy sigma-algebra thus the elements of Q are in G .

The operations in Q' are the same as in G , and are defined in the same way. If Q' is not closed then, it must be completed until it becomes closed. Thus, Q is the minimal closed subalgebra in G containing Q' . In the cases interesting for us we shall see that $Q = Q'$.

Q is named the quantified algebra of G .

4. Quantified algebras

In this section we give some statements concerning the quantified algebras of different fuzzy algebras.

Theorem. Let G be an R -fuzzy sigma-algebra. Then its quantified algebra Q is a Boolean algebra.

Proof. For proving the statement, let us assume, that $X = \{x_0\}$ has only one element. Let K and L be elements of G such that

$$L = \forall K = \bigwedge_{i=1}^{\infty} K.$$

Then

$$L = K \wedge \bigwedge_{i=2}^{\infty} K = K \wedge \forall K = K \wedge L.$$

Thus

$$L = K \wedge L.$$

Thus, because of axiom 4 in II, either $K = T = \neg \emptyset$ or $L = \emptyset$, (And then $K = \emptyset$ too.) In the first case, if $K = T$, then $L = T$. Therefore, taking an arbitrary K, L can be only \emptyset or T .

Similarly, if $M = \exists K$, M is \emptyset or T .

In the case when $X = \{x_0\}$, it can be easily proved now, that Q contains only two elements T and \emptyset , since

$$T \vee T = T, T \vee \emptyset = T, \emptyset \vee \emptyset = \emptyset; \quad T \wedge T = T, T \wedge \emptyset = \emptyset, \emptyset \wedge \emptyset = \emptyset;$$

$$\neg T = \emptyset, \neg \emptyset = T.$$

Let us consider now the general case, where X is an arbitrary set. Then for an arbitrary $x \in X$,

$$qL(x) = q\emptyset(x_0) \text{ or } qT(x) \quad \text{and} \quad qM(x_0) = q\emptyset(x_0) \text{ or } qT(x_0).$$

Considering now the ordinary sets and their characteristic functions over X , an isomorphism to the set of all possible functions $qL(x)$ and $qM(x)$, i.e., all membership functions of the elements can be found. The isomorphism mapping orders to one characteristic function the membership function which has the value qT where it is 1, and $q\emptyset$ where it is 0.

Finally, Q has not to be completed, which fact can be proved from the ideas concerning the connection between K and L , and K and M , used in the above part of the proof.

As the characteristic functions of the subsets of a set X form a Boolean algebra, the isomorphic Q is Boolean, too.

Theorem. Let G be an S -algebra, such that cardinality of S is finite. Then the quantified algebra Q of G , is Boolean.

Proof can be found in [8].

For the further examination we introduce some more definitions:

Definition. Let G be a fuzzy sigma-algebra. Then G is named (weakly) sigma-associative, if

$$\bigvee_{i=1}^{\infty} (A_i \vee B_i) = \bigvee_{i=1}^{\infty} A_i \vee \bigvee_{i=1}^{\infty} B_i,$$

and

$$\bigwedge_{i=1}^{\infty} (A_i \wedge B_i) = \bigwedge_{i=1}^{\infty} A_i \wedge \bigwedge_{i=1}^{\infty} B_i, \quad \text{for all } A_i, B_i \in G.$$

Definition. If G is an S -algebra which is sigma-algebra and sigma-associative then we call it an SA -algebra.

Theorem. Let G be an SA -algebra. Then the quantified algebra Q of G is Boolean.

First, we prove two dual lemmata.

Lemma. If $qA > qB$, then $q\exists A \cong q\exists B$.

Proof. Because of axiom group 4 in III,

$$qA \vee A > qB \vee B.$$

Similarly,

$$q(A \vee A) \vee (A \vee A) > q(B \vee B) \vee (B \vee B),$$

etc. etc.

$$q \bigvee_{i=1}^{\infty} A \cong q \bigvee_{i=1}^{\infty} B,$$

i.e.

$$q\exists A \cong q\exists B,$$

which had to be proved.

Lemma. If $qA > qB$, then $q\forall A \cong q\forall B$.

The proof is the dual of the above one.

Now, we return to the proof of the original statement.

Let A be an arbitrary element different from \emptyset and T , and $B = A \vee A$, $C = \exists A$. (Since G is a sigma-algebra thus $C \in G$.)

Then, obviously,

$$qB \cong qC, \quad \text{and, since } \exists \exists A = \exists A = C,$$

using the first lemma, we obtain

$$q\exists B = q \bigvee_{i=1}^{\infty} B \cong qC.$$

Using the property of sigma-associativity, this can be written in the following form

$$q \bigvee_{i=1}^{\infty} A \vee \bigvee_{i=1}^{\infty} A = qC \vee C \cong qC.$$

Because of axiom group 4 in III, $qC \vee C \cong qC$, which means, that

$$C = C \vee C.$$

Restricting our considerations to sets X with one single element, similarly as in the former proof, it can be seen, that C can be only T or \emptyset . That means, that if allowing arbitrary sets X , the elements of Q are in isomorphism with the ordinary subsets of X (the dual proof for $D = \forall A$ must, as a matter of course, added to the above one), thus Q is a Boolean algebra.

Without going into details, we mention, that some related definitions (strong sigma-associativity and sigma-continuity) are dealt with in [8]; on the basis of them different theorems can be proved, which give some practical conditions for a fuzzy algebra to be sigma-associative, etc.

Here we give only one statement yet, referring to Zadeh's so-called non-interactive algebra.

Theorem. If G is a non-interactive fuzzy sigma-algebra, (i.e. a distributive lattice), the quantified algebra Q of G is identical to G . (The very simple proof can also be found in [8].)

Finally, we should stress again the fact, that — because of the examinations in Section 2 — all the above theorems hold for arbitrary n -fuzzy structures, as well.

5. On the application possibilities

Here, we do not intend to deal with application possibilities in detail, only some very general aspects will be given.

Let us assume, that we have a concrete representant of an — e.g. R -fuzzy — algebra, as an example, we present here the most simple and well-known one: the so-named "soft definitions", as follows

$$qA \vee B = qA + qB - qAqB,$$

$$qA \wedge B = qAqB,$$

$$q \neg A = 1 - qA.$$

(A much more general discussion of representants can be found in [8].)

We intend to use this fuzzy calculus for realizing learning algorithms. A very obvious way for this is the formulation of a learning fuzzy automaton which acts on behalf of a membership function (or more than one functions) representing a present strategy for the automaton. The membership function itself is gained by a

learning generative process, where fuzzy calculus has a basic role. We have a system that we want to control by our automaton. In a particular situation (element of X , now the situation space) the automaton produces a particular output. (Let us assume now, that the automaton is the fuzzy generalization of a bang-bang type controller, i.e., it has two possible outputs, 1 and 0, and — being a fuzzy one — something like “meanthings” between this two, the values of the interval $[0, 1]$.)

It can be proved in the case of some special optimality criteria, that the best output is the maximal possible value (or the minimal) in a particular situation. Then a given output of the automaton can be more or less near to this or, in contrary, more or less far from it. In the first case a little “quantum” of membership function will be “added” (i.e. disjoined) to the given function such that it converges to the maximal value, and in the second case, it is added in such a way that it converges to the other extremum. Then, if the learning process is going on, it can be hoped, that the values of the membership function(s) over all x -es will be near to the optimal extremum.

Now, we do not intend to deal with the problem, how a fuzzy automaton executes a fuzzy instruction. Let us assume, that both possible outputs have one membership function, that we hope to converge to 1 over such x 's where an output 1 or 0 is needed, respectively; and to 0, where 0 or 1, respectively. (Thus two functions belong to the outputs “1” and “0”, respectively.)

The real mathematical problem is here: Whether this algorithm converges in the exact mathematical sense?

Theorem. If C is a calculus representing G , an R -fuzzy algebra, the disjunction in C is \vee , and the inverse operation of it (which can be defined on the basis of axiom group 5 in II) is denoted by \vee^- , f is an indicator taking the value 1, if a strategy S is “good” (in any sense), and 0, if it is “bad”, then the following algorithm generating the membership function $qS(x)$ of the strategy S is convergent over all x , where the expected value of f is greater than 0

$$qS(x)_0 = 0,$$

$$qS'(x)_{n+1} = \text{if } f = 1 \text{ then } qS'(x) \vee q(x) \text{ else } qS'(x) \vee^- q(x),$$

$$qS(x)_{n+1} = \text{if } qS'(x)_n \cong 0 \text{ then } qS'(x)_n \text{ else } 0.$$

$q(x)$ is chosen such that $q(x)$ is greater than a positive ε and less than 1 over x , and 0 elsewhere.

Here, we do not prove this statement.

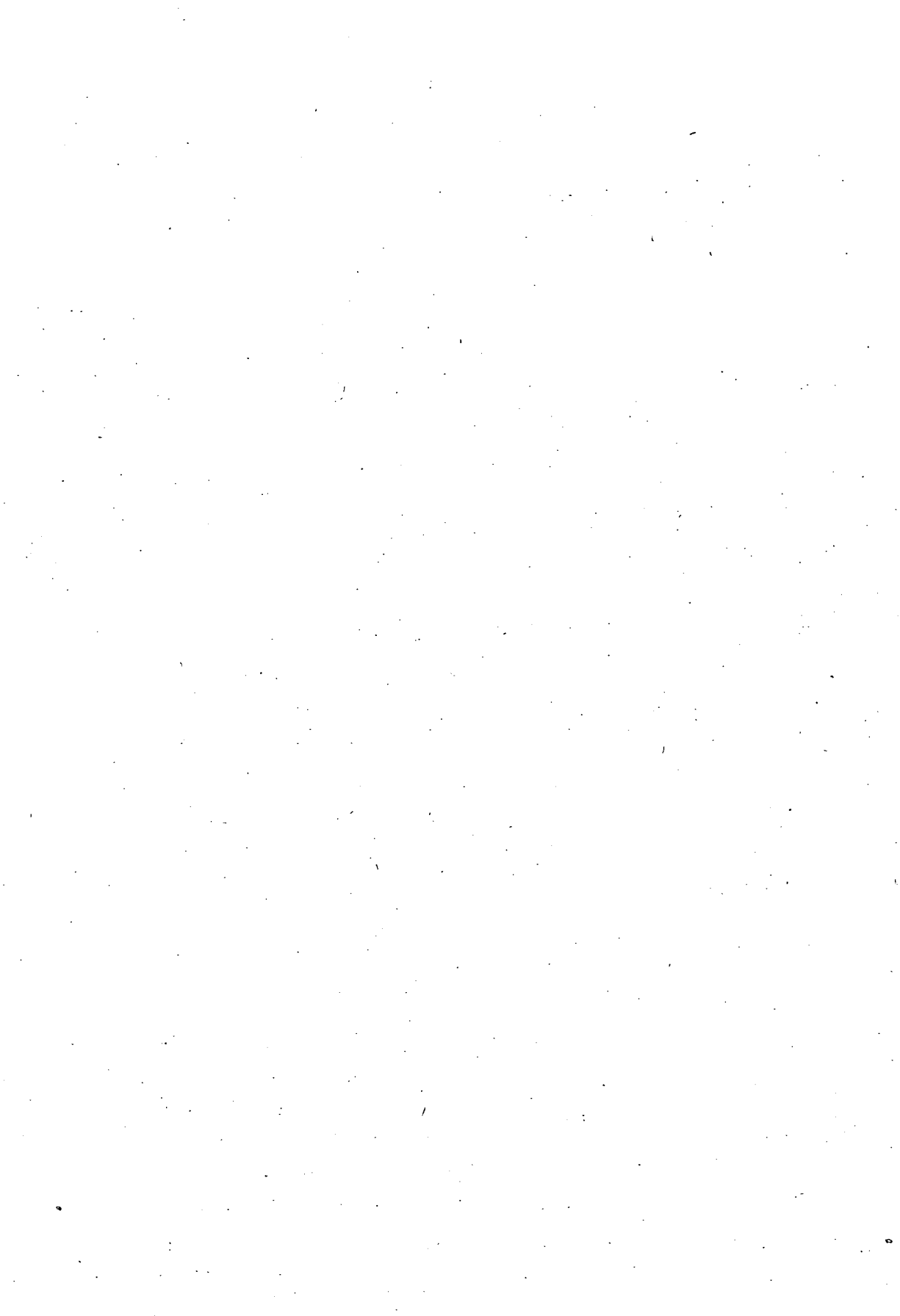
Using other types of $q(x)$, similar theorems can be proved (however, in the case of not so general conditions). The same is the case, when G is not R -fuzzy, but SA -algebra. Then, however, the algorithm must slightly be changed.

Finally, we mention, that we have always some experimental results using an algorithm similar to the above one (controlled is a stochastic, double integrating system).

References

- [1] BELLMAN, R. É., R. KALABA, L. A. ZADEH, Abstraction and pattern classification, *J. Math. Anal. Appl.*, v. 13, 1966, pp. 1—7.
- [2] ZADEH, L. A., Fuzzy sets, *Information and Control*, v. 8, 1965, pp. 338—353.
- [3] KÓCZY, L. T., A fuzzy halmazok néhány elméleti kérdése (Some theoretical questions of fuzzy of sets), *Report, Dept. of Proc. Contr.*, Tech. Univ. Budapest, Jan. 1974.
- [4] KÓCZY, L. T., M. HAJNAL, A new fuzzy calculus and its application as a pattern recognition technique, Modern Trends of Cybernetics and General Systems, *Proc. of the 3rd Congress of WOGSC*, Bucharest, Aug. 1975, pp. 66—81.
- [5] KÓCZY, L. T., A fuzzy halmazok elmélete és műszaki alkalmazásai (The theory of fuzzy sets and their applications in technology), *Report, Dept. of Proc. Contr.*, Tech. Univ. Budapest, May 1975.
- [6] HAMACHER, H., Über logische Verknüpfungen unscharfer Aussagen und deren zugehörige, Bewertungsfunktionen, Arbeitsbericht, *Lehrst. für Unternehmensforschung RWTH Aachen*, July 1975.
- [7] RÖDDER, W., On “and” and “or” connective in fuzzy set theory, *WP Lst. für Unt.forsch. RWTH Aachen*, Jan. 1975.
- [8] KÓCZY, L. T., Fuzzy algebrák és műszaki alkalmazásaik néhány kérdése (Fuzzy algebras and some questions of their technical applications), *PhD thesis, Dept. of Proc. Contr.*, Tech. Univ. Budapest, May 1976.
- [9] ZADEH, L. A., The concept of linguistic variable and its application to approximate reasoning I, *Information Sci.*, v. 8, 1975, pp. 199—249.
- [10] SELYE, H., *From dream to discovery on being a scientist*, McGraw Hill Book Co.; 2nd Hung. ed. Akadémiai Kiadó, Budapest, 1974.
- [11] MIZUMOTO, M. & K. TANAKA, Some properties of fuzzy sets of type 2, *Information and Control.*, v. 31, 1976, pp. 312—340.

(Received Feb. 16, 1977)



On the formal definition of VDL-objects

By I. FEKETE and L. VARGA

Originally the VDL (Vienna Definition Language) was designed for defining programming languages [1], [2], [3], but recently it has been used as a general technique of defining data structures and algorithms [4].

The VDL is a definition system. This system consists of objects, a machine operating on objects and a programming language.

The VDL-objects are abstractions of data structures of a certain type. In this paper we deal with the objects and the basic operators of VDL manipulating on objects.

The VDL-objects form a set with the elements of which there are associated selection and construction operators. The basic properties of the operators are taken as axioms and their main properties are proved. A complete formal system of VDL-objects is given, which can be regarded as a detailed elaboration of the axiomatic definition of VDL data structures given in [4] and [5].

Definition 1. The elements of the non empty set OB are called objects, if there exists a finite set S of selectors and a construction function k such that

$$s:OB \rightarrow OB \text{ for all } s \in S, \text{ and}$$

$$k:OB \times S \times OB \rightarrow OB.$$

It is assumed the validity of the following:

Axiom 1. If $t \in OB$, $s \in S$, $t_1 \in OB$, then

$$s(k(t, s, t_1)) = t_1,$$

and

$$s'(k(t, s, t_1)) = s'(t) \text{ for all } s' \in S \text{ and } s' \neq s.$$

The "fixed point" of the system, i.e. the null object of the set OB is defined as follows:

Definition 2. A $t \in OB$ is called the *null object* if and only if

$$(\forall s \in S)(s(t) = t)$$

Axiom 2. There is exactly one null object.

In the following we denote the null object by Ω .

The objects can be classified according to their "distance" from the null object. The so called elementary objects are "nearest" to the null object, and they can be defined in the following way:

Definition 3. A $t \in OB$ is called *elementary object* if and only if

$$(\forall s \in S)(s(t) = \Omega).$$

Let EO be the set of elementary objects.

Definition 4. The elements of the set

$$CO = OB \setminus EO$$

are called *composite objects*.

Axiom 3. If $t \in OB$ then there exists an integer N_t such that for any sequence $s_1 \in S, s_2 \in S, \dots, s_n \in S, (n \geq N_t)$

$$s_n(\dots(s_2(s_1(t))\dots)) = \Omega.$$

COROLLARY 1. There is no $t \in OB, t \neq \Omega$ for which

$$s_m(\dots(s_2(s_1(t))\dots)) = t.$$

Axiom 4. Elementary objects are regarded as different, that is if

$$EO = \{\dots, t_i, \dots, t_j, \dots\}$$

then $t_i \neq t_j$.

Definition 5. The objects $t_1 \in CO$ and $t_2 \in CO$ are *equal* if and only if

$$(\forall s \in S)(s(t_1) = s(t_2)).$$

Lemma 1. Ω is an elementary object.

Proof. This results from Definitions 2 and 3.

Theorem 1. If EO has at least two elements, then CO is a non empty set.

Proof. Let $t \in OB, t \neq \Omega$ and $s \in S$. Then by Axiom 1.

$$s(k(t, s, t)) = t,$$

and hence

$$k(t, s, t) \in CO.$$

Theorem 2. If CO is a non empty set, then EO has at least two elements.

Proof. Let us suppose, that $EO = \{\Omega\}$. Let $t \in CO$. Then, by definition,

$$(\exists s_1 \in S)(s_1(t) \neq \Omega).$$

But the set EO has only one element. Therefore

$$t_1 = s_1(t) \in CO.$$

Hence

$$(\exists s_i \in S)(s_2(t_1) \neq \Omega).$$

The repeated application of this procedure leads to a contradiction to Axiom 3.

Now let us consider the "structure" of the objects. First of all we define the immediate components of the object.

Definition 6. If $t \in OB$ and $s \in S$, then the object $s(t)$ is called the *immediate component* of the object t .

COROLLARY 2. All immediate components of an elementary object are the null object.

Definition 7. Let $t \in CO$. The *immediate characteristic set* of t is defined as

$$\{\langle s_1 : s_1(t) \rangle, \langle s_2 : s_2(t) \rangle, \dots, \langle s_m : s_m(t) \rangle\}$$

where

$$s_i(t), (s_i \in S), \quad i = 1, 2, \dots, m$$

are all non null immediate components of the object t .

Lemma 2. Any composite object can be uniquely represented with its immediate characteristic set.

Proof. This follows from Definition 5 immediately.

Definition 8. Let $t \in CO$ and let

$$t_1, t_2, \dots, t_m$$

be every non null immediate component of the object t such that

$$s_i(t) = t_i, \quad i = 1, 2, \dots, m,$$

then the symbol

$$\mu_0(\langle s_1 : t_1 \rangle, \langle s_2 : t_2 \rangle, \dots, \langle s_m : t_m \rangle)$$

will stand for the object t .

By Lemma 2, this is an unambiguous representation of the object t .

The composite object can be represented also by a tree as shown in Fig. 1.

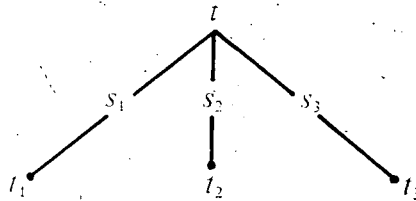


Fig. 1

The tree of the object

$$t = \mu_0(\langle s_1 : t_1 \rangle, \langle s_2 : t_2 \rangle, \langle s_3 : t_3 \rangle)$$

Theorem 3. If

$$t = \mu_0(\langle s_1: t_1 \rangle, \langle s_2: t_2 \rangle, \dots, \langle s_m: t_m \rangle)$$

then the object t can be constructed from the objects t_1, t_2, \dots, t_m by applying the operation k m times.

Proof. Let us consider the following sequence

$$y_1 = k(\Omega, s_1, t_1),$$

$$y_2 = k(y_1, s_2, t_2),$$

$$\vdots$$

$$y_m = k(y_{m-1}, s_m, t_m).$$

Then, by Axiom 1, we have

$$s_m(y_m) = t_m,$$

$$s_{m-1}(y_m) = s_{m-1}(y_{m-1}) = t_{m-1},$$

$$\vdots$$

$$s_1(y_m) = s_1(y_{m-1}) = \dots = s_1(y_1) = t_1,$$

and for every $s \in S, s \neq s_i, i = 1, 2, \dots, m,$

$$s(y_m) = \Omega.$$

Hence, by Lemma 2, we have the result $y_m = t$.

Definition 9. The composition

$$\chi = s_m \circ s_{m-1} \circ \dots \circ s_1, \quad s_i \in S, \quad i = 1, 2, \dots, m$$

is called *composite selector*. The result of applying a composite selector χ to an object $t \in CO$ is defined as follows

$$\chi(t) = s_m(\dots(s_2(s_1(t)))) \dots).$$

Let S^* be the set of all the composite selectors constructed by the elements of S and all the simple selectors.

Definition 10. If $\chi = s \circ \chi', \chi' \in S^*, s \in S, t \in OB, t_1 \in OB$ then

$$k(t, \chi, t_1) = k(t, \chi', k(\chi'(t), s, t_1)).$$

Theorem 4. The objects $t_1 \in CO, t_2 \in CO$ are equal if and only if

$$(\forall \chi \in S^*)(\chi(t_1) = \chi(t_2)).$$

Proof. Let $\chi = s_m \circ \dots \circ s_2 \circ s_1$.

If $t_1 = t_2$, then, by Definition 5, we have

$$\begin{aligned} s_1(t_1) &= s_2(t_2) \\ s_2 \circ s_1(t_1) &= s_2 \circ s_1(t_2) \\ &\vdots \\ \chi(t_1) &= \chi(t_2) \end{aligned}$$

for every composite selector in S^* .

On the other hand, if

$$(\forall \chi \in S^*)(\chi(t_1) = \chi(t_2))$$

then

$$(\forall s \in S)(s(t_1) = s(t_2)),$$

i.e., the immediate characteristic sets of t_1 and t_2 are equal. Therefore, $t_1 = t_2$.

Theorem 5. For every object $t \in CO$ there exists a composite selector χ such that

$$\chi(t) \in EO \setminus \{\Omega\}.$$

Proof. By Definition 4, we have

$$(\exists s_1 \in S)(s_1(t) \neq \Omega).$$

If $t_1 = s_1(t) \in EO$, then the assertion follows immediately. Otherwise $t_1 \in CO$ and as above we have

$$(\exists s_2 \in S)(s_2(t_1) = t_2 \neq \Omega),$$

and so forth, by virtue of Axiom 3,

$$(\exists n \geq 1)(s_n(t_{n-1}) \in EO \setminus \{\Omega\}).$$

Therefore, for $\chi = s_n \circ \dots \circ s_2 \circ s_1$,

$$\chi(t) \in EO \setminus \{\Omega\}.$$

Definition 11. Let

$$H_1(t) = \{\langle s_1 : t_1 \rangle, \langle s_2 : t_2 \rangle, \dots, \langle s_m : t_m \rangle\}$$

be the immediate characteristic set of $t \in CO$. Let us introduce the notation

$$H_1(t) = \{\langle \chi_1^{(1)} : t_1^{(1)} \rangle, \langle \chi_2^{(1)} : t_2^{(1)} \rangle, \dots, \langle \chi_m^{(1)} : t_m^{(1)} \rangle\},$$

where

$$\chi_i^{(1)} = s_i, t_i^{(1)} = t_i, \quad i = 1, 2, \dots, m.$$

If

$$(\exists j, 1 \leq j \leq m)(t_j^{(1)} \in CO)$$

choose the smallest j such that $t_j^{(1)} \in CO$ and let

$$\bar{H}(t_i) = \{\langle s'_1 : z_1 \rangle, \langle s'_2 : z_2 \rangle, \dots, \langle s'_n : z_n \rangle\}$$

be the immediate characteristic set of t_i , where $s'_i \in S, i = 1, 2, \dots, n$.

Substituting $\bar{H}(t_i)$ into $H_1(t)$, the following set may be derived

$$\begin{aligned} H_2(t) &= \{\langle \chi_1^{(1)} : t_1^{(1)} \rangle, \dots, \langle s'_1 \circ \chi_i^{(1)} : z_1 \rangle, \dots, \langle s'_n \circ \chi_i^{(1)} : z_n \rangle, \dots, \langle \chi_m^{(1)} : t_m^{(1)} \rangle\} = \\ &= \{\langle \chi_1^{(2)} : t_1^{(2)} \rangle, \dots, \langle \chi_M^{(2)} : t_M^{(2)} \rangle\}. \end{aligned}$$

Iterating the preceding procedure, we can generate the sequence of sets

$$H_1(t), H_2(t), H_3(t), \dots$$

The elements of this sequence are called *characteristic sets* of t .

Definition 12. Let $t \in CO$ and let

$$H_i(t) = \{\langle \chi_1^{(i)} : t_1^{(i)} \rangle, \dots, \langle \chi_{m_i}^{(i)} : t_{m_i}^{(i)} \rangle\}, \quad i = 1, 2, \dots$$

be all the characteristic sets of t . The characteristic set

$$H_N(t) = \{\langle \chi_1^{(N)} : t_1^{(N)} \rangle, \dots, \langle \chi_{m_N}^{(N)} : t_{m_N}^{(N)} \rangle\}$$

for which

$$(\forall j, 1 \leq j \leq m_N)(t_j^{(N)} \in EO)$$

is called the *elementary characteristic set* of t .

Theorem 6. Let $t \in CO$, then the sequence of characteristic sets

$$H_1(t), H_2(t), \dots$$

is finite, and its last element is the elementary characteristic set of t .

Proof. By Axiom 3, the procedure given in Definition 11 terminates after a finite number of steps, and the last element of the sequence obviously satisfies the criteria of the elementary characteristic set of t .

Theorem 7. A composite object t can be uniquely represented with its any characteristic set.

Proof. On the base of the procedure given in Definition 11, by Lemma 2, Theorem 7 follows for t and $H_1(t)$. Similarly it is also true for t_i and $\bar{H}(t_i)$. Hence t can be uniquely represented with $H_2(t)$. Similarly we may show that t can be uniquely represented with $H_3(t)$, $H_4(t)$, ...

COROLLARY 3. It follows from the Theorems 6 and 7 that any $t \in CO$ can be unambiguously represented by an elementary characteristic set.

Definition 13. Let $t \in CO$ and let

$$H(t) = \{\langle \chi_1 : t_1 \rangle, \dots, \langle \chi_m : t_m \rangle\}$$

be a characteristic set of t . Then the object t can be notified by the symbol

$$\mu_0(\langle \chi_1 : t_1 \rangle, \dots, \langle \chi_m : t_m \rangle).$$

Based on Theorem 6, every composite object can be represented by a tree in which there are only elementary objects as terminal nodes. For example the

composite object

$$t = \mu_0(\langle s_1 : a \rangle, \langle s_1 \cdot s_2 : b \rangle, \langle s_3 \cdot s_2 : c \rangle)$$

where

$$\{a, b, c\} \subseteq EO$$

can be represented by the tree shown in Fig. 2.

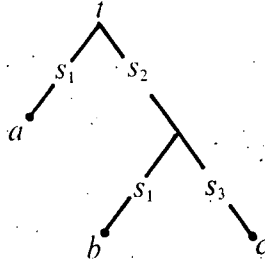


Fig. 2

The tree of the object

$$t = \mu_0(\langle s_1 : a \rangle, \langle s_1 \circ s_2 : b \rangle, \langle s_3 \circ s_2 : c \rangle)$$

Definition 14. A composite selector χ is said to be dependent on a composite selector χ' if and only if

$$\chi' = \chi'' \circ \chi \quad \text{or} \quad \chi' = \chi \quad \text{for some} \quad \chi'' \in S^*.$$

Definition 15. The selectors χ_i and χ_j are said to be independent if and only if neither χ_i is dependent on χ_j nor χ_j is dependent on χ_i .

Theorem 8. Let

$$H(t) = \{\langle \chi_1 : t_1 \rangle, \langle \chi_2 : t_2 \rangle, \dots, \langle \chi_m : t_m \rangle\}$$

be a characteristic set of $t \in CO$. Then for all $1 \leq i, j \leq m, i \neq j$ implies that χ_i and χ_j are independent.

Proof. The proof is by induction on k in $H_k(t)$. Based on definition 14, every pair $\chi_i^{(1)}, \chi_j^{(1)}$ is independent in $H_1(t)$.

Assume that Theorem 8 holds for every $H_i(t), 1 \leq i \leq k$ and prove it for $H_{k+1}(t)$. Let

$$H_k(t) = \{\langle \chi_1^{(k)} : t_1^{(k)} \rangle, \dots, \langle \chi_{m_k}^{(k)} : t_{m_k}^{(k)} \rangle\},$$

and $t_1^{(k)} \in EO, \dots, t_{j-1}^{(k)} \in EO, \text{ but } t_j^{(k)} \in CO.$ Let

$$\{\langle s_1 : z_1 \rangle, \dots, \langle s_N : z_N \rangle\}$$

be the immediate characteristic set of $t_j^{(k)}$. Then

$$H_{k+1}(t) = \{\langle \chi_1^{(k)} : t_1^{(k)} \rangle, \dots, \langle s_1 \circ \chi_j^{(k)} : z_1 \rangle, \dots, \langle s_N \circ \chi_j^{(k)} : z_N \rangle, \dots, \langle \chi_{m_k}^{(k)} : t_{m_k}^{(k)} \rangle\}$$

Here, by our assumption, every pair $\chi_p^{(k)}, \chi_q^{(k)}$ is independent. We now have to show that

$$\chi_p = s_p \circ \chi_j^{(k)} \quad (1 \leq p \leq N)$$

and

$$\chi_q = s_q \circ \chi_j^{(k)} \quad (1 \leq q \leq N)$$

($p \neq q$) are also independent. For example, we can easily see, that there exists no χ such that

$$\chi_p = \chi \circ \chi_q,$$

because

$$s_p \neq \chi \circ s_q.$$

Similarly, it is also easy to show that any $s_p \circ \chi_j^{(k)}$ is not dependent on $\chi_i^{(k)}$, $i \neq j$.

Theorem 9. If $\chi \in S^*$, $t \in OB$, $t_1 \in OB$, then

$$\chi(k(t, \chi, t_1)) = t_1$$

and

$$\chi'(k(t, \chi, t_1)) = x'(t)$$

provided that χ is not dependent on χ' , $\chi' \in S^*$.

Proof. We prove the theorem by induction. Consider the selector

$$\chi_m = s_m \circ s_{m-1} \circ \dots \circ s_1 \quad (s_i \in S).$$

If $\chi = \chi_1$ and $\chi' = \chi_1$ then the Theorem is true by Axiom 1.

The principle of the induction states:

If our Theorem holds for any $\chi = \chi_k$ and $\chi' = \chi'_j$ with $1 \leq k, j \leq m$ then it also holds for any $\chi = \chi_k$ and $\chi' = \chi'_j$ with $1 \leq k, j \leq m+1$.

Assume the Theorem is true for all

$$\chi = \chi_k \quad \text{and} \quad \chi' = \chi'_j \quad \text{with} \quad 1 \leq k, j \leq m$$

and prove it for all $1 \leq k, j \leq m+1$. By Definition 10,

$$k(t, \chi_{m+1}, t_1) = k(t, \chi_m, k(\chi_m(t), s, t_1)),$$

where $\chi_{m+1} = s \circ \chi_m$. Furthermore, by our assumption,

$$s \circ \chi_m(k(t, \chi_m, k(\chi_m(t), s, t_1))) = s(k(\chi_m(t), s, t_1)),$$

and, by Axiom 1,

$$s(k(\chi_m(t), s, t_1)) = t_1.$$

Consider the second equation in the Theorem. If χ is not dependent on χ' then

$$\chi' = \bar{\chi}' \circ s', \quad \chi = \bar{\chi} \circ s, \quad \text{where} \quad s \neq s'$$

or

$$\chi' = \chi'_1 \circ \chi_2, \quad \chi = \chi_1 \circ \chi_2$$

where

$$\chi'_1 = s'_1 \circ s'_2 \circ \dots \circ s'_i,$$

$$\chi_1 = s_1 \circ s_2 \circ \dots \circ s_j$$

and

$$s'_i \neq s_j.$$

Hence

$$\bar{\chi}' \circ s'(k(t, \chi, t_1)) = \bar{\chi}' \circ s'(k(t, s, k(s(t), \bar{\chi}, t_1))) = \bar{\chi}' \circ s'(t)$$

or

$$\begin{aligned} \chi'_1 \circ \chi_2(k(t, \chi, t_1)) &= \chi'_1 \circ \chi_2(k(t, \chi_2, k(\chi_2(t), \chi_1, t_1))) = \\ &= \chi'_1(k(\chi_2(t), \chi_1, t_1)) = \chi'_1 \circ \chi_2(t). \end{aligned}$$

This completes the proof.

Theorem 10. Let

$$t = \mu_0(\langle \chi_1 : t_1 \rangle; \dots, \langle \chi_m : t_m \rangle).$$

Then the object t can be constructed from the objects t_1, t_2, \dots, t_m by applying the operation k m times.

Proof. The proof is analogous to that of Theorem 3.

Due to this Theorem, every composite object can be constructed from elementary objects too. Hence each composite object is a structure of elementary objects.

Definition 16.

$$k(t, \chi_1, t_1, \chi_2, t_2, \dots, \chi_n, t_n) = k(k(t, \chi_1, t_1), \chi_2, t_2, \dots, \chi_n, t_n)$$

Theorem 11. If χ_1 and χ_2 are independent, then for arbitrary objects t_1 and t_2

$$k(t, \chi_1, t_1, \chi_2, t_2) = k(t, \chi_2, t_2, \chi_1, t_1).$$

Proof. Consider the right side of the equation. It follows from Theorem 9 that

$$\chi_1(k(k(t, \chi_2, t_2), \chi_1, t_1)) = t_1,$$

and for every χ' which is not dependent on χ_1 ,

$$\chi'(k(k(t, \chi_2, t_2), \chi_1, t_1)) = \chi'(k(t, \chi_2, t_2)).$$

Hence for $\chi' = \chi_2$,

$$\chi_2(k(k(t, \chi_2, t_2), \chi_1, t_1)) = \chi_2(k(t, \chi_2, t_2)) = t_2,$$

and for every χ'' which is not dependent on χ_2 ,

$$\chi''(k(k(t, \chi_2, t_2), \chi_1, t_1)) = \chi''(k(t, \chi_2, t_2)) = \chi''(t).$$

Similarly, it can be shown that

$$\chi_1(k(t, \chi_1, t_1, \chi_2, t_2)) = t_1,$$

$$\chi_2(k(t, \chi_1, t_1, \chi_2, t_2)) = t_2,$$

and for every $\bar{\chi}$ which is not dependent on χ_1 and χ_2

$$\bar{\chi}(k(t, \chi_1, t_1, \chi_2, t_2)) = \bar{\chi}(t).$$

This completes the proof.

In the VDL the following notation is used:

$$\mu(t; \langle \chi_1 : t_1 \rangle, \dots, \langle \chi_n : t_n \rangle) \equiv k(t, \chi_1, t_1, \dots, \chi_n, t_n).$$

О формальном определении VDL-объектов

Первоначально VDL был предназначен для определения языков программирования [1, 2, 3], но в последнее время применяется и как общий метод определения структуры данных и алгоритмов [4].

VDL является системой определения. Эта система состоит из объектов, машины десит-вующей над объектами, и из языка программирования.

VDL-объекты представляют собой структуры данных определенного типа. В данной работе изучаются объекты и основные VDL-операторы, действующие над объектами.

С элементами множества VDL-объектов связаны операторы выбора и конструирования. Основные свойства этих операторов изложены в виде аксиом, а дальнейшие свойства доказаны. Таким образом, задана полная формальная система VDL-объектов, которую можно рассматривать как подробную разработку аксиоматического определения структуры данных VDL, предложенного в [4] и [5].

EÖTVÖS LORÁND UNIVERSITY
H-1088 BUDAPEST, HUNGARY
MÚZEUM KRT. 6—8.

CENTRAL RESEARCH INSTITUTE FOR PHYSICS
H-1525 BUDAPEST, HUNGARY
BOX 49.

References

- [1] LUCAS, P., P. LAUER, H. STIEGLEITNER, Method and notation for the formal definition of programming languages, *IBM Lab. Vienna*, TR 25.087, 1968.
- [2] LUCAS, P., K. WALK, On the formal description of PL/1, *Annual Review in Automatic Programming*, v. 6, 1969, pp. 105—182.
- [3] NEUHOLD, E. J., The formal description of programming languages, *IBM Systems J.*, v. 10, 1971.
- [4] LEE, J. A. N., *Computer semantics*, Van Nostrand Reinhold Company, New York, 1972.
- [5] WEGNER, P., The Vienna definition language, *Comput. Surveys*, v. 4, 1972, pp. 5—63.

(Received March 12, 1977)

An effective theorem proving algorithm

By P. ECSEDI—TÓTH and A. VARGA

Summary

A simple procedure is presented which is a sound and complete calculus for first-order logic. The method can easily be implemented on computers and it has considerable advantage in manual work.

Introduction

Nowadays computer science becomes more and more 'scientific' after its 'naive' age of the last 30 years. Much work has been devoted to create 'software industry', however, for although there is a shortage of really effective methods.

Research of artificial intelligence gives rise to a hope of solving this problem. First program-correctness-proving procedures were suggested by Turing, Floyd, Manna and others in their works on 'modeling' intelligent processes of human thinking. The soul of these procedures is a pure mathematical logical tool: a theorem-proving algorithm, or in other words an automatic deduction.

After the activity of Gödel, Skolem, Turing, and Herbrand [5, 9, 10, 6], J. A. Robinson proposed an elegant practical realization of automatic deduction in [8]. Since 1965, when the first paper on Robinson's resolution principle appeared, many theorem-proving algorithms, each of them is an 'improvement' of the resolution principle, have been published. However, almost all these methods have two common disadvantages. Namely, they have a deep combinatorial nature and one has to do a not-so-few unnecessary steps in pre-procedure.

The input of a theorem-prover, working on the basis of a version of the resolution principle, is a formula in the first-order logic. (First-order logic is one of the most important tools for 'software industry', but theorem-provers of other types of languages e.g. zero- or second-order logic, non-classical (modal) logic, fuzzy logic and other multi-valued logics etc. are also known.) This input formula has to be in prenex normal form, in disjunctive normal form and in Skolem-normal form. In this paper we present a procedure which can be applied to formulae not being in disjunctive normal form (skolemization and prenex normal form are needed).

Our method has some similarity, although it is essentially different from that, to the resolution principle. Our procedure is a 'valuation' or rather a 'computation of logical value' of the given formula, and thus it has less combinatorial character.

Some other resolution-like algorithms eliminating the steps needed to get a disjunctive normal form of the given input formula, are also known [7], [2, 3] but our method over the 'computation' property, mentioned above, differs in the following points, as well. It offers:

- i) Easier treating of quantified variables;
- ii) More effectiveness thanks to the less combinatorial nature of our method;
- iii) Easier manual and also machine execution.

In the first section the notions, notations and facts needed for the development are surveyed.

The second section deals with the essence of the material. In the third one the procedure is defined for first-order logic. Completeness theorems are only stated, the long but not too complicated proofs will appear elsewhere.

§ 1. Preliminaries

By a type τ we mean an ordered quintuple

$$\tau = \langle I, J, K, t', t'' \rangle$$

where I, J, K are pairwise disjoint sets (sets of relation symbols, function symbols and constant symbols, respectively); t', t'' are functions for which

$$\begin{aligned} \text{Domain } (t') &= I, \\ \text{Range } (t') &\subset \omega, \\ \text{Domain } (t'') &= J, \\ \text{Range } (t'') &\subset \omega. \end{aligned}$$

The set of τ -type formulae of zero- and first-order (defined in the standard way) will be denoted by

$${}_0F^\tau \text{ and } {}_1F^\tau.$$

\mathfrak{S}^τ is the class of τ -type algebraic structures (the class of possible models). Let $M(\varphi)$ denote the following class of structures

$$M(\varphi) = \{\mathfrak{A} \mid \mathfrak{A} \in \mathfrak{S}^\tau, \mathfrak{A} \models \varphi\}.$$

$\varphi \in {}_1F^\tau$ is a tautology (unsatisfiable) iff

$$M(\varphi) = \mathfrak{S}^\tau \quad (M(\varphi) = \emptyset).$$

The notion of algorithm is used in a naive sense. We shall suppose that a finite set of symbols Z is fixed.

Let

$$Z^* = \{\langle z_1, \dots, z_n \rangle \mid n < \omega, z_i \in Z \text{ if } i \leq n\}.$$

By the definition of an algorithm we mean an element of Z^* .

If $z \in Z^*$ then there exists a partial function

$$\tilde{z}: Z^* \rightarrow Z^*,$$

the meaning of z . Consequently, if $z_1, z_2, z_3 \in Z^*$ then

$$\tilde{z}_1(z_2) = z_3$$

means that the result of applying the algorithm z_1 to the algorithm z_2 is the algorithm z_3 .

We will assume that the elements of ${}_1F^*$ and ω are represented in Z^* (for example ${}_1F^* \subset Z^*$ and $\omega \subset Z^*$).

Let $z \in Z^*$ and $A \subset Z^*$. z enumerates the set A iff

$$\text{Domain } (\tilde{z}) = \omega,$$

$$\text{Range } (\tilde{z}) = A.$$

Suppose there exist two distinguished elements in Z^* , namely, z_0 and z_1 which represent the truth-values TRUE and FALSE, respectively.

If $z \in Z^*$, $A \subseteq Z^*$, then z decides the set A iff

$$\text{Domain } (\tilde{z}) = Z^*,$$

$$\text{Range } (\tilde{z}) = \{z_0, z_1\}$$

and for any $z' \in Z^*$,

$$\tilde{z}(z') = z_0 \text{ iff } z' \in A.$$

§ 2. Deciding the set of zero-order sentences

For an arbitrary zero-order sentence φ , let S_φ be a sequence of prime sentences (relation symbols with no variables in their argumentum places) occurring in φ .

Let φ be a zero-order sentence. The following algorithm z defines a binary tree $\tilde{z}(\varphi)$ of φ :

Step 1. Start with the sentence φ as initial vertex.

Step 2. Choose the first not used element of S_φ , say p , where ψ is the actually treated vertex, and substitute the truth-values TRUE and FALSE, respectively, for p in ψ . Draw two edges, labelled by p and its negation, respectively, from the vertex.

Step 3. Apply the following rules to obtain the truth-value-free version of the substituted vertex ψ or a single truth-value along both of the edges: for any zero-order sentence φ ,

$$\varphi \wedge 1 \leftrightarrow \varphi; \quad 1 \wedge \varphi \leftrightarrow \varphi \quad (1)$$

$$\varphi \wedge 0 \leftrightarrow 0; \quad 0 \wedge \varphi \leftrightarrow 0 \quad (2)$$

$$\varphi \vee 1 \leftrightarrow 1; \quad 1 \vee \varphi \leftrightarrow 1 \quad (3)$$

$$\varphi \vee 0 \leftrightarrow \varphi; \quad 0 \vee \varphi \leftrightarrow \varphi \quad (4)$$

$$\varphi \rightarrow 1 \leftrightarrow 1 \quad (5)$$

$$\varphi \rightarrow 0 \leftrightarrow \bar{\varphi} \quad (6)$$

$$1 \rightarrow \varphi \leftrightarrow \varphi \quad (7)$$

$$0 \rightarrow \varphi \leftrightarrow 1 \quad (8)$$

$$(\varphi \leftrightarrow 1) \leftrightarrow \varphi; \quad (1 \leftrightarrow \varphi) \leftrightarrow \varphi \quad (9)$$

$$(\varphi \leftrightarrow 0) \leftrightarrow \bar{\varphi}; \quad (0 \leftrightarrow \varphi) \leftrightarrow \bar{\varphi} \quad (10)$$

$$\varphi \wedge \varphi \leftrightarrow \varphi \quad (11)$$

$$\varphi \wedge \bar{\varphi} \leftrightarrow 0; \quad \bar{\varphi} \wedge \varphi \leftrightarrow 0 \quad (12)$$

$$\varphi \vee \varphi \leftrightarrow \varphi \quad (13)$$

$$\varphi \vee \bar{\varphi} \leftrightarrow 1; \quad \bar{\varphi} \vee \varphi \leftrightarrow 1 \quad (14)$$

$$\varphi \rightarrow \varphi \leftrightarrow 1 \quad (15)$$

$$(\varphi \leftrightarrow \varphi) \leftrightarrow 1 \quad (16)$$

$$\varphi \leftrightarrow \bar{\bar{\varphi}} \quad (17)$$

$$\bar{1} \leftrightarrow 0; \quad \bar{0} \leftrightarrow 1 \quad (18)$$

where \wedge , \vee , \rightarrow , \leftrightarrow , $\bar{}$, 0 and 1 is the symbol for conjunction, disjunction, implication, equivalence, negation, truth-values FALSE and TRUE, respectively.

The truth-value-free formulae or single truth-values, obtained from the actually treated vertex will be the two new vertices χ and η . Let the sequence of zero-order prime sentences of χ (η) be S_χ (S_η) with the same order as in S_φ .

Note that $S_\chi = S_\psi - \{p\}$ ($S_\eta = S_\psi - \{p\}$).

Step 4. Do steps 2 and 3 for the vertices χ and η if they are not single truth-values. If each of the vertices newly made is a truth-value, the procedure terminates.

The vertices which are truth-values will be called final-vertices.

Example 1. Assume that the given zero-order sentence is

$$\varphi = \overline{A(a, b) \wedge \overline{C}} \rightarrow ((B(a) \vee C) \leftrightarrow A(a, b))$$

where

$$t'(A) = 2, \quad t'(B) = 1, \quad t'(C) = 0, \quad t''(a) = 0, \quad t''(b) = 0.$$

The tree produced by the algorithm is indicated in Fig. 1, and its "computation" in Fig. 2.

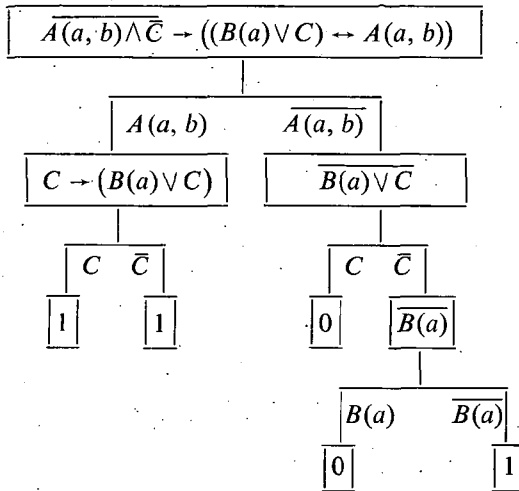


Fig. 1

Let ${}_0f^{dec}$, ${}_0g^{dec}$ be algorithms such that

i) ${}_0f^{dec} : {}_0F^v \rightarrow \{z_0, z_1\};$

ii) ${}_0g^{dec} : {}_0F^v \rightarrow \{z_0, z_1\},$

$${}_0f^{dec}(\varphi) = \begin{cases} z_0 & \text{if all the final-vertices of } \tilde{z}(\varphi) \text{ are } 1 \\ z_1 & \text{otherwise,} \end{cases}$$

$${}_0g^{dec}(\varphi) = \begin{cases} z_0 & \text{if all the final-vertices of } \tilde{z}(\varphi) \text{ are } 0 \\ z_1 & \text{otherwise.} \end{cases}$$

Note that ${}_0f^{dec}$ and ${}_0g^{dec}$ do exist, i.e., the final vertices of $\tilde{z}(\varphi)$ do not depend on the order of S_φ .

Theorem. [11].

- i) ${}_0f^{dec}$ decides the set of zero-order tautologies;
- ii) ${}_0g^{dec}$ decides the set of zero-order unsatisfiable sentences.

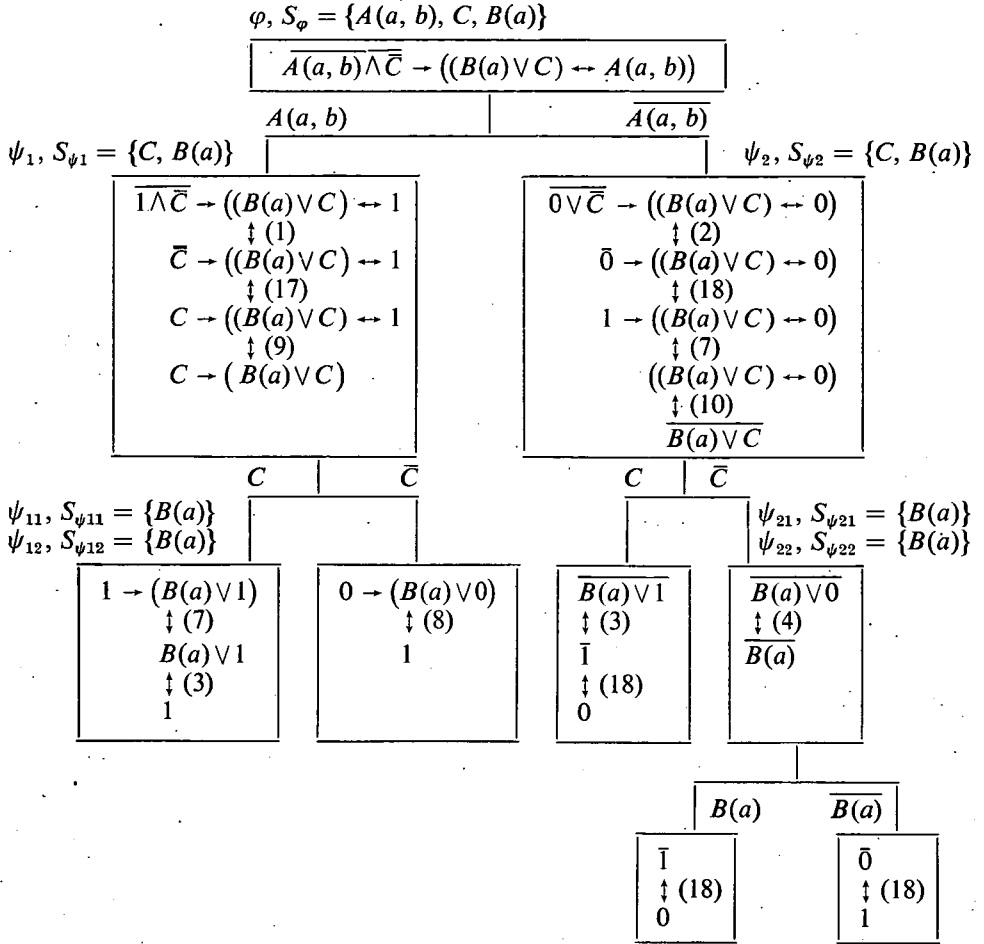


Fig. 2

§ 3. Tree-constructing for first-order sentences

We define the well-known Skolem-extension of type τ in the following way:

$$\tau^{(0)} = \tau.$$

If

$$\tau^{(m)} = \langle I, J^{(m)}, K, t', t''^{(m)} \rangle \text{ is defined}$$

then

$$\tau^{(m+1)} = \langle I, J^{(m)} \cup J^*, K, t', t''^{(m+1)} \rangle$$

where

$$J^* = \{f_\varphi | \varphi(x_1, \dots, x_n) \in_1 F^{\tau^{(m)}}\}$$

and f_φ is a new function symbol for every $\varphi(x_1, \dots, x_n) \in {}_1F^{\tau(m)}$ and

$$t''^{(m+1)}(f_\varphi) = n - 1;$$

furthermore,

$$t''^{(m+1)}(f) = t''^{(m)}(f)$$

if $f \notin J^*$.

$$\tau^s = \cup \{\tau^{(m)} \mid m < \omega\}.$$

Theorem (Skolem). There exist algorithms ex and un such that

i)
$$\tilde{ex}: {}_1F^\tau \rightarrow {}_1F^{\tau^s},$$

$$\tilde{un}: {}_1F^\tau \rightarrow {}_1F^{\tau^s};$$

ii) For any $\varphi \in {}_1F^\tau$, φ is in prenex form,

$\tilde{ex}(\varphi)$ does not contain universal quantifiers,

$\tilde{un}(\varphi)$ does not contain existential quantifiers;

iii) For $\varphi \in {}_1F^\tau$ in prenex form

$$M(\varphi) = \mathfrak{S}^\tau \quad \text{iff} \quad M(\tilde{ex}(\varphi)) = \mathfrak{S}^{\tau^s},$$

$$M(\varphi) = \emptyset \quad \text{iff} \quad M(\tilde{un}(\varphi)) = \emptyset.$$

This theorem is a simple consequence of the Skolem Normal Form Theorem [1, 4, 9].

Let a sentence φ be given in prenex form. Consider the formulae $\tilde{ex}(\varphi)$ and $\tilde{un}(\varphi)$.

We define the sets P_φ^{ex} (P_φ^{un}) and S_φ^{ex} (S_φ^{un}) as follows:

P_φ^{ex} (P_φ^{un}) is the sequence of zero-order prime formulae of $\tilde{ex}(\varphi)$ ($\tilde{un}(\varphi)$) which are not sentences i.e., contain at least one free variable. The prime formulae χ and η differing only in free variables are distinguished in P_φ^{ex} (P_φ^{un}).

S_φ^{ex} (S_φ^{un}) is the sequence of zero-order prime sentences of $\tilde{ex}(\varphi)$ ($\tilde{un}(\varphi)$) if there exist such sentences. In the opposite case, substitute arbitrary constant symbols for the free variables of the elements of P_φ^{ex} (P_φ^{un}) and let the substituted formulae, which are now sentences, be in the sequence S_φ^{ex} (S_φ^{un}).

Example 2. Let the formula φ be

$$(\forall x)(\forall y)(\forall z)[(E(z, x) \leftrightarrow E(z, y)) \leftrightarrow I(x, y)].$$

Then

$$\tilde{un}(\varphi) = \varphi,$$

$$\tilde{ex}(\varphi) = [E(a, b) \leftrightarrow E(a, c)] \leftrightarrow I(b, c),$$

$$P_\varphi^{un} = \langle E(z, x), E(z, y), I(x, y) \rangle,$$

$$S_\varphi^{un} = \langle E(a, a), I(a, a) \rangle$$

for any arbitrary constant symbol a ,

$$P_\varphi^{ex} = \emptyset,$$

$$S_\varphi^{ex} = \langle E(a, b), E(a, c), I(b, c) \rangle.$$

Denote the matrices of $\tilde{e}x(\varphi)$ and $\tilde{u}n(\varphi)$ by $\tilde{e}x(\varphi)^*$ and $\tilde{u}n(\varphi)^*$, respectively.

We recall the concept of unification [8]. Let φ and ψ be prime-formulae. φ and ψ are unifiable iff there exists a substitution such that the substituted φ and ψ are identical literal by literal.

Let φ be a first-order sentence in prenex form. The following generalization of z defines two binary trees $\tilde{z}^{ex}(\varphi)$ and $\tilde{z}^{un}(\varphi)$:

Step 1. Start with the formula $\tilde{e}x(\varphi)^* (\tilde{u}n(\varphi)^*)$ to obtain the initial vertex of $\tilde{z}^{ex}(\varphi)$ (and $\tilde{z}^{un}(\varphi)$), respectively.

Step 2. Choose the first element of $S_\psi^{ex} (S_\psi^{un})$, say p , where ψ is the actually treated vertex and substitute p first by 1 then by 0. Draw two edges from ψ , labelled by p and its negation, respectively.

Seek for the elements of $P_\psi^{ex} (P_\psi^{un})$ which can be unified with p , and execute the unifying substitution for all such elements.

Step 3. Apply the tautologies listed in §2 as rules (1)—(18) to obtain the truth-value-free version of the substituted vertex or to obtain a single truth-value along both edges. These truth-value-free formulae or the single truth-values will be the two new vertices χ and η . Then construct the sequences $S_\chi^{ex} (S_\chi^{un}), S_\eta^{ex} (S_\eta^{un}), P_\chi^{ex} (P_\chi^{un}), P_\eta^{ex} (P_\eta^{un})$.

Step 4. Repeat the steps for every vertices newly obtained if they are not single truth-values. If all of the vertices are truth-values the procedure terminates.

Note that $\tilde{z}^{ex}(\varphi)$ and $\tilde{z}^{un}(\varphi)$ exist for any first-order sentence φ if it is in prenex form. The trees do not depend on the order of S_φ^{ex} and S_φ^{un} . If φ is a tautology or it is unsatisfiable then $\tilde{z}^{ex}(\varphi)$ and $\tilde{z}^{un}(\varphi)$ are finite trees.

Example 3. Let us construct the trees $\tilde{z}^{ex}(\varphi)$ and $\tilde{z}^{un}(\varphi)$ for the formula of Example 2. They are indicated by Fig. 3 and Fig. 4.

Let ${}_1f^{com}, {}_1g^{com}$ be algorithms such that

$$\begin{aligned} \text{i)} \quad & {}_1f^{com}: {}_1F^{rs} \rightarrow Z^*, \\ & {}_1\tilde{g}^{com}: {}_1F^{rs} \rightarrow Z^*, \end{aligned}$$

where elements of ${}_1F^{rs}$ are supposed to be in prenex normal form.

ii)

$$\begin{aligned} {}_1f^{com}(\varphi) &= \begin{cases} z_0 & \text{if all the final vertices of } \tilde{z}^{ex}(\varphi) \text{ are 1,} \\ \text{undefined} & \text{otherwise,} \end{cases} \\ {}_1\tilde{g}^{com}(\varphi) &= \begin{cases} z_0 & \text{if all the final vertices of } \tilde{z}^{un}(\varphi) \text{ are 0.} \\ \text{undefined} & \text{otherwise.} \end{cases} \end{aligned}$$

Remark. ${}_1f^{com}, {}_1g^{com}$ exist by the previous note.

Theorem. Let $\varphi \in {}_1F^{rs}$ be a sentence in prenex form. Then

- i) φ is a tautology iff ${}_1f^{com}(\varphi) = z_0$,
- ii) φ is unsatisfiable iff ${}_1\tilde{g}^{com}(\varphi) = z_0$.

$$P_{\varphi}^{un} = \langle E(z, x), E(z, y), I(x, y) \rangle$$

$$S_{\varphi}^{un} = \langle E(a, a), I(a, a) \rangle$$

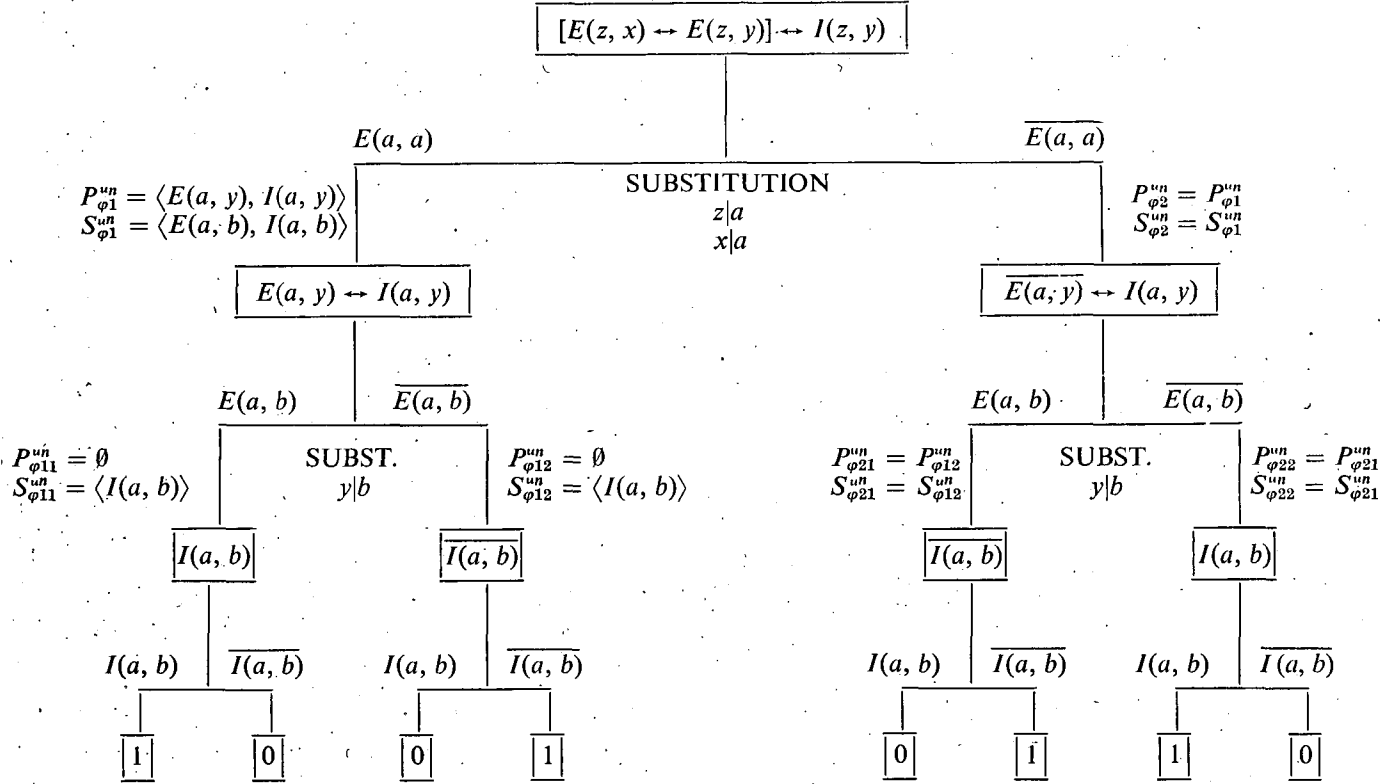


Fig. 3
The tree $\bar{z}^{un}(\varphi)$ of φ

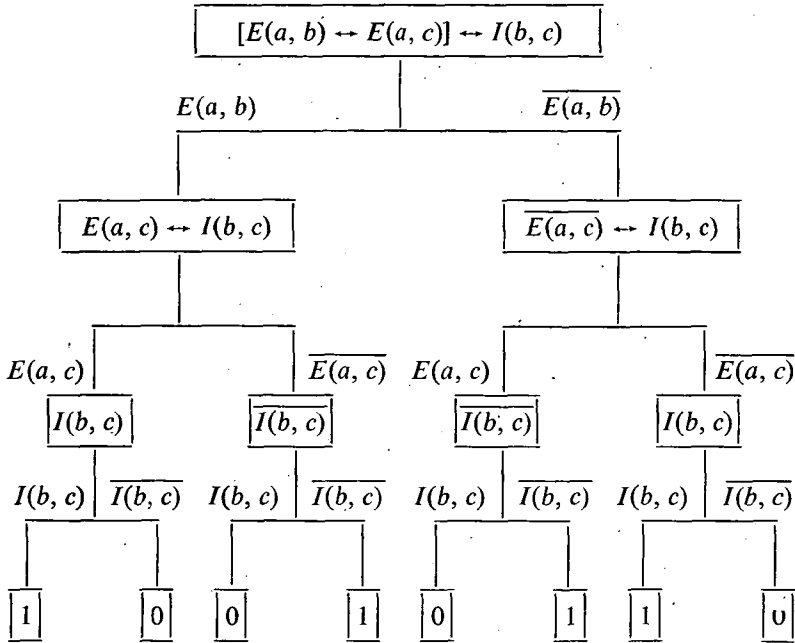


Fig. 4

The tree $\tilde{z}^{ex}(\varphi)$ of φ

Example 4. Let the following sentences be given:

$$\varphi_1: (\forall x)(A(x) \rightarrow (B(x) \wedge C(x))),$$

$$\varphi_2: (\exists x)(A(x) \wedge D(x)),$$

$$\psi: (\exists x)(D(x) \wedge C(x)).$$

Assume we want to know whether ψ is a consequence of φ_1 and φ_2 . There are two possibilities: to show that the formula $\varphi' = \varphi_1 \wedge \varphi_2 \rightarrow \psi$ is a tautology; or to show that the formula $\varphi'' = \varphi_1 \wedge \varphi_2 \wedge \neg \psi$ is unsatisfiable. The previous theorem ensures us that if the formula φ' is a tautology then ${}_1f^{com}(\varphi') = z_0$; and if the formula φ'' is unsatisfiable then ${}_1\tilde{g}^{com}(\varphi'') = z_0$.

$$\begin{aligned} \tilde{e}x(\varphi') &= (\exists x)(\exists y)([A(x) \rightarrow (B(x) \wedge C(x))] \wedge \\ &\quad \wedge [A(a) \wedge D(a)] \rightarrow [D(y) \wedge C(y)]), \end{aligned}$$

$$\begin{aligned} \tilde{u}n(\varphi'') &= (\forall x)(\forall y)([A(x) \rightarrow (B(x) \wedge C(x))] \wedge \\ &\quad \wedge [A(a) \wedge D(a)] \wedge [\neg D(y) \vee \neg C(y)]), \end{aligned}$$

where a is a Skolem-function (constant).

The tree $\tilde{z}^{ex}(\varphi')$ is indicated in Fig. 5.

All the final-vertices of the tree $\tilde{z}^{ex}(\varphi')$ are 1. Thus ${}_1f^{com}(\varphi') = z_0$, i.e., φ' is a tautology. The tree $\tilde{z}^{un}(\varphi'')$ is very similar to $\tilde{z}^{ex}(\varphi')$ but all the final-vertices are 0. Thus ${}_1\tilde{g}^{com}(\varphi'') = z_0$, i.e. φ'' is unsatisfiable.

$$P_{\varphi'}^{\text{ex}} = \langle A(x), B(x), C(x), D(y), C(y) \rangle$$

$$S_{\varphi'}^{\text{ex}} = \langle A(a), D(a) \rangle$$

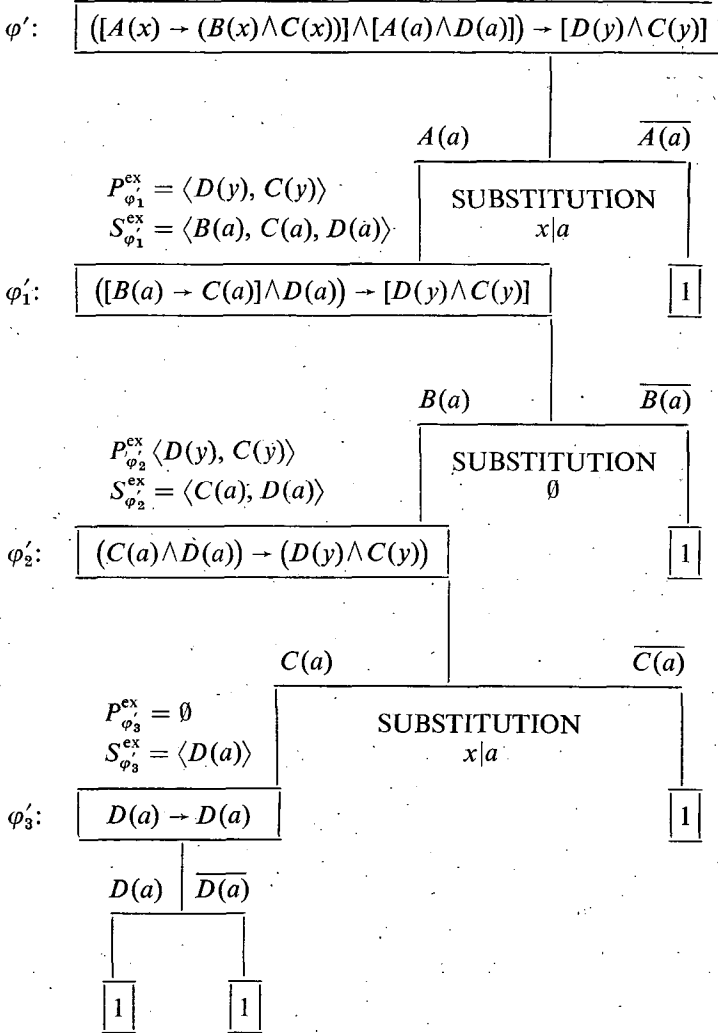


Fig. 5

The tree $\tilde{z}^{\text{ex}}(\varphi')$

RESEARCH GROUP ON MATHEMATICAL LOGIC AND THEORY OF AUTOMATA OF THE HUNGARIAN ACADEMY OF SCIENCES H-6720 SZEGED, HUNGARY SOMOGYI U. 7.

BOLYAI INSTITUTE OF THE ATTILA JÓZSEF UNIVERSITY H-6720 SZEGED, HUNGARY ARADI VÉRTANÚK TERE 1.

References

- [1] ANDRÉKA, H., T. GERGELY, I. NÉMETHI, Easily comprehensible mathematical logic and its model theory, *KFKI memo*, mimeographed, Budapest, 1975.
- [2] CHANG, C. L., Theorem proving by generation of pseudo-semantic trees, *Div. of Comput. Res. and Technology*, National Institute of Health, Bethesda, Maryland, 1971.
- [3] CHANG, C. L., Theorem proving with variable-constrained resolution, *Information Sci.*, v. 4, 1972, pp. 217—231.
- [4] CHANG, C. C., H. J. KEISLER, *Model Theory*, North-Holland Publ. Co., Amsterdam, 1973.
- [5] GÖDEL, K., Die Vollständigkeit der Axiome des logischen Funktionenkalküls, *Monatsh. Math. Phys.*, v. 37, 1930, pp. 349—360.
- [6] HERBRAND, J., Recherches sur la theorie de la demonstration, *Travaux de la Societe des Sciences et des Lettres de Varsovie*, v. 33, 1930, p. 128.
- [7] PRAWITZ, D., Advances and problems in mechanical proof procedures, *Machine Intelligence*, v. 4, (edited by B. Meltzer and D. Michie), American Elsevier, New York, 1969, pp. 59—71.
- [8] ROBINSON, J. A., A machine oriented logic based on the resolution principle, *J. Assoc. Comput. Mach.*, v. 1965, pp. 23—41.
- [9] SKOLEM, T., Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit oder Beweisbarkeit mathematischer Sätze nebst einem Theorem über dichte Mengen, *Skrifter utgitt av Videnskapselskapet i Kristiania, I, Mat. Naturv. Kl.*, v. 4, 1920, p. 36.
- [10] TURING, A. M., Computing machinery and intelligence, *Mind*, v. 59, 1950, pp. 433—460.
- [11] VARGA, A., P. ECSEDI-TÓTH, F. MÓRICZ, An effective method for minimizing Boolean polynomials, *Research Report*, Research Group on Mathematical Logic and Theory of Automata of the Hungarian Academy of Sciences, 1977, p. 40.

(Received May 17, 1977)

INTERCELLAS

an interactive cellular space simulation language

By T. LEGENDI

Abstract

Intercellas is an integral member of a family of languages for cellular space simulation and programming. It has been designed for interactive use on mini-computers and gives possibilities for simulation of heterogeneous spaces, too.

I. Design goals

The main goals of the research and the reasons to define a new family of cellular space simulation and programming languages [8, 10, 12] rather than using the existing ones [1, 2, 3, 4] — are described in detail in [11].

The CELLAS cellular space simulation and programming language family has been designed for medium size computers and mainly for use under batch operating systems. In many cases it seems more favourable and economic to use a smaller but interactive system.

The instruction set of INTERCELLAS is compatible with CELLAS except dynamic program editing and interrupt possibilities.

It is natural that after detecting syntactic or semantic errors control is passed to the programmer.

Some instructions of the language are more simple, than their equivalents in CELLAS, but combined with interactive facilities. For example: instruction ON, that monitors continuously the space, in the original language has wider condition description part and has action part, too. In INTERCELLAS it has no action part, preprogrammed automatic reaction is impossible, the only action is that control is passed to the programmer at the consol (display).

The instructions of the language are command-type. They are interpreted and executed as they enter the processor. The structure of the language is very simple and natural:

- a) to begin a simulation the programmer first needs tools to define topology, neighbourhood, transition function(s), size of the space and initial configuration should be formed;

- b) control of the simulation steps is the central goal of the language;
- c) the display of the results should be flexible to support obtaining as much information as possible but in compact form through controlling the time, mode, origin and destination of the results to display;
- d) interactive intervention into the simulation process;
- e) other utilities;
- f) as minicomputer configurations are very different (peripherals, mass storage, core memory) and the structure of the language is simple, it is very natural to associate a system generation feature (actual peripherals, instructions, limits for tables — including transition function table, the maximum size of the cellular space and other internal tables, etc — may determine actual processors).

In the following the instruction set of the language will be briefly discussed and shown that it meets the above basic requirements.

The instruction format is simple, assembly type. The instructions begin with instruction code (may be abbreviated or full) which is followed by parameters. The instructions may be labeled in the usual way.

II. Groups of the instruction set

1. Definition of the space. The *topology* of the space may be defined through assigning to each cell its neighbours' relative coordinates.

Transition functions may be defined by a list of terms, by tables, by definition of terms in a tree form, or by feature definition and new state assignment instructions.

Examples: *Table form 1* 0110 1001 1001 0110 ... (table of summa mod 2). Here $n_0 + 2n_1 + 4n_2 + \dots$ defines an address in the table where the new state should be found ($n_0, n_1, n_2 \dots$ are the current states of the cell and its neighbours).

Table form 2 (Summa mod 4)

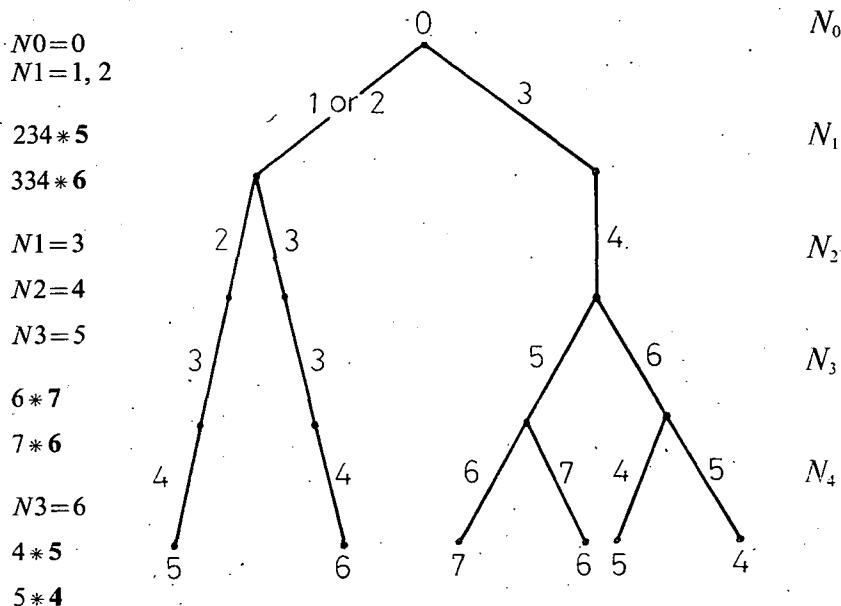
<i>T</i>	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

Here $T(T(T(T(n_0, n_1), n_2), n_3), n_4))$ assigns the next state.

List of terms

old state	neighbours states	new state
0	1234	5
0	1334	6
0	2234	5
0	2334	6
0	3456	7
0	3457	6
0	3464	5
0	3465	4

Tree form (for the above terms)



Another term form:

0	-0+1+2-3-4	*	5
old state	neighbourhood description		new state

where in the simplest case + means there is at least one neighbour in the state following the plus sign, and - means that there is no neighbour in the state following the minus sign.

A more sophisticated interpretation may be used by *feature definition instructions*, their execution assigns a feature to *i* and after it, + *i* means that the neighbourhood has the *i*th feature, - *i* means the opposite.

Examples for possible features:

- a) the third neighbour is in state 3 or
 the third neighbour is in state 4 or
 the second neighbour is in state 3
- b) there are no neighbours in state 2
- c) the number of neighbours in state 2 and state 4 is even
 or the number of neighbours in state 3 is 2

Transition functions may be read from files or libraries, too. (See I/O instructions).

Dimension and actual *size* of the space may be directly defined, a space may have *dummy* cells on the *boundary* or may be *closed* in form of a circle in one dimension (or *torus* in two dimension).

To *different parts* of the space *different* (predefined) *transition functions* may be assigned.

Initial configuration. Initial configurations may be defined by a set of simple geometrical instructions: rectangles may be filled with the same value (state), lines (of the same value) may be specified, configurations may be copied from parts of the space to other parts of it and configurations may be shifted from outside of the space. I/O instructions (see below) may also define configurations in the space.

2. Input-output. Under I/O we mean here the flow of main (characteristic) data structures e.g. transition functions and configurations.

The instructions control the flow (and implicitly the conversion) of data among files, core memory and libraries.

On a *file* data are in a usual line form and may be accessed only sequentially.

In the *memory* data are in internal representation.

In a *library* data are in compressed formats, they may be accessed in associative way.

Simple library handling instructions are also included (library initiation/purge, listing, deleting objects from the library etc.).

3. Simulation. The central instruction of the language effects execution of n steps of simulation (the cellular space should be defined prior to the execution of the simulation). As the simulation is sequential and therefore slow a look ahead algorithm is applied to speed up processing. Cells are grouped in two classes:

- a) *closed cells* (neither the cell nor its neighbour cells had changed their previous states during the last step of simulation)
- b) *open cells* (either the cell itself or any of its neighbour cells had changed its state during the last step of simulation)

Naturally, the transition function is computed only for open cells. The cells are made open or closed during each simulation step. The presumed status is closed and it is made open only if a change occurs in the state of the cell or in the states of its neighbours. The status is stored in two independent flag-bits (open/closed, next open/next closed) which alternate during the consecutive simulation steps. The main characteristic of the algorithm is that no operation is performed on the closed cells—neither on their states nor on their flag—bits.

4. Display of the results. It is possible to define

- a) *when* — in which steps of the simulation
- b) *from where* — from what parts of the cellular space
- c) *how* — which characters correspond to the states of the cells (conversion) to display the results. Steps for display are indicated by a set of stored display/do not display instructions. (They are stored in the order of execution.) They effect in parallel — each of them defines a set of steps (non-negative integers) in form $a+bx+c^y$ $x, y=1, 2, 3, \dots$. If an actual simulation step does not fall in any set or it is a member of at least one do not display set — no display will take place. The result will be displayed if the actual step occurs (only) in a display set.

For example

	a	b	c
DISPLAY	1	0	0
DISPLAY	2	0	0
DISPLAY	0	0	3
DO NOT DISPLAY	27		

Results will be displayed in the first, second and each $3n$, $n=1, 2, 3, \dots$ steps excluding 27. (1, 2, 3, 9, 81, ...)

DISPLAY	0	1	0
DO NOT DISPLAY	0	3	0

Results will be displayed in each step excluding each third step. (1, 2, 4, 5, 7, 8, 10, ...)

Destination of the results may also be programmed, this is treated in the next section.

5. Flow of information. It is possible to designate files (including peripheral equipments) for

- a) the program input file,
- b) program output files,
- c) result output files.

The program input file should contain an INTERCELLAS program or part of it.

The program output file(s) will contain the executed and (during execution) skipped instructions. This file will contain only syntactically correct instructions.

The result output file(s) will contain the result of the run.

All these files may be dynamically designated. (For example the source program may be read from two or more files).

Files may be designated for more than one purpose (in a meaningful way).

A natural way is to read program from the card reader or the console and to print the program and the results in an intermixed form e.g. each instruction is followed by its result (if any).

Another possibility is to make selective output: for example

- a) to print results and send them to a mass storage file, not to print program;
- b) to read program from console, to punch program, to print results and program and so forth.

6. Flow of control. Normally instructions are executed sequentially. It is possible to skip instructions conditionally or unconditionally with an optional input file change.

(There are two main reasons to combine skipping with input file change:

- a) skip instructions typed in at the main periphery have no meaning without a change of the input file;
- b) in some situations at the time of a change of the input file skipping may be needed. For example a prepared program on cards or tape during its run passes control to the programmer. After executing the needed intervention the programmer wants to pass control back to the program — but may be not to the point where it was interrupted).

It is possible to stop the work of the processor finally, or to begin a new simulation.

Continuous monitoring of the cellular space against simple conditions (whether the state of the (i, j) cell is S) may be programmed. When the condition is met, control is passed to the programmer. Conditions may be cleared, too.

7. Interactivity. There is a simple editing possibility: characters are specified for deleting consecutive characters or parameters or a whole line typed in previously. (The read process is character oriented rather than line oriented.)

A longer (many steps) simulation process may be interrupted by sending a special interrupt character. The actual simulation step is finished and control is passed to the main periphery. After the execution of arbitrary instructions it is possible to return to the previous activity and input periphery. Two special characters ensure the return. One of them selects to continue the interrupted simulation, while the other skips the remaining steps and execution continues by interpreting the next instruction on the previous input periphery.

Interrupt may be used in case of any problem with an actual (when it is not the main) periphery — the change to the input periphery ensures intervention preserving the previous state of the system.

8. System generation instructions. The list of *actual peripherals* defines the needed/unneeded handlers and tables.

The *main peripheral* (which should be a read/write peripheral, it has priority and interrupt may be initiated only from it) may be designated. The actual *set of instructions* may be defined including the definition of the names of the instructions. In connection with this there is an implicit possibility for *selecting* transition function evaluating *table search procedures* and this selection may be done explicitly, too.

There is a possibility to limit some explicit and implicit data structures namely the *maximum size* of the cellular *space*, and *transition function* tables; the *maximum number* of parallel *ON conditions*, *DISPLAY/DO NOT DISPLAY conditions* etc.

Error messages may be defined too.

An automatic user's manual generation and autotest generation will also be incorporated into this group of instructions which is implemented in FORTRAN IV as cross-software for the simulator.

III. Implementation, emulators

INTERCELLAS has been implemented on minicomputers CII—10010 (subset only), TPA (equivalent to PDP—8) and R—10 (equivalent to MITRA—15). The processors are coded in assembly and FORTRAN—IV languages.

For speeding up the simulation special firmware (cellular space emulator) has been designed for 2 state and 16 state spaces.

The main goal of the project is to design cellular processors: emulators will serve partly as their working models. INTERCELLAS has been designed as a software tool for simulation, which may be interpreted as machine code level programming of cellular processors (with added utilities which are important and useful but do not increase the machine code level). In this way INTERCELLAS may be used for testing cellular processors and developing higher level languages for them.

RESEARCH GROUP ON MATHEMATICAL LOGIC
AND THEORY OF AUTOMATA OF THE
HUNGARIAN ACADEMY OF SCIENCES
H-6720 SZEGED, HUNGARY
SOMOGYI U. 7.

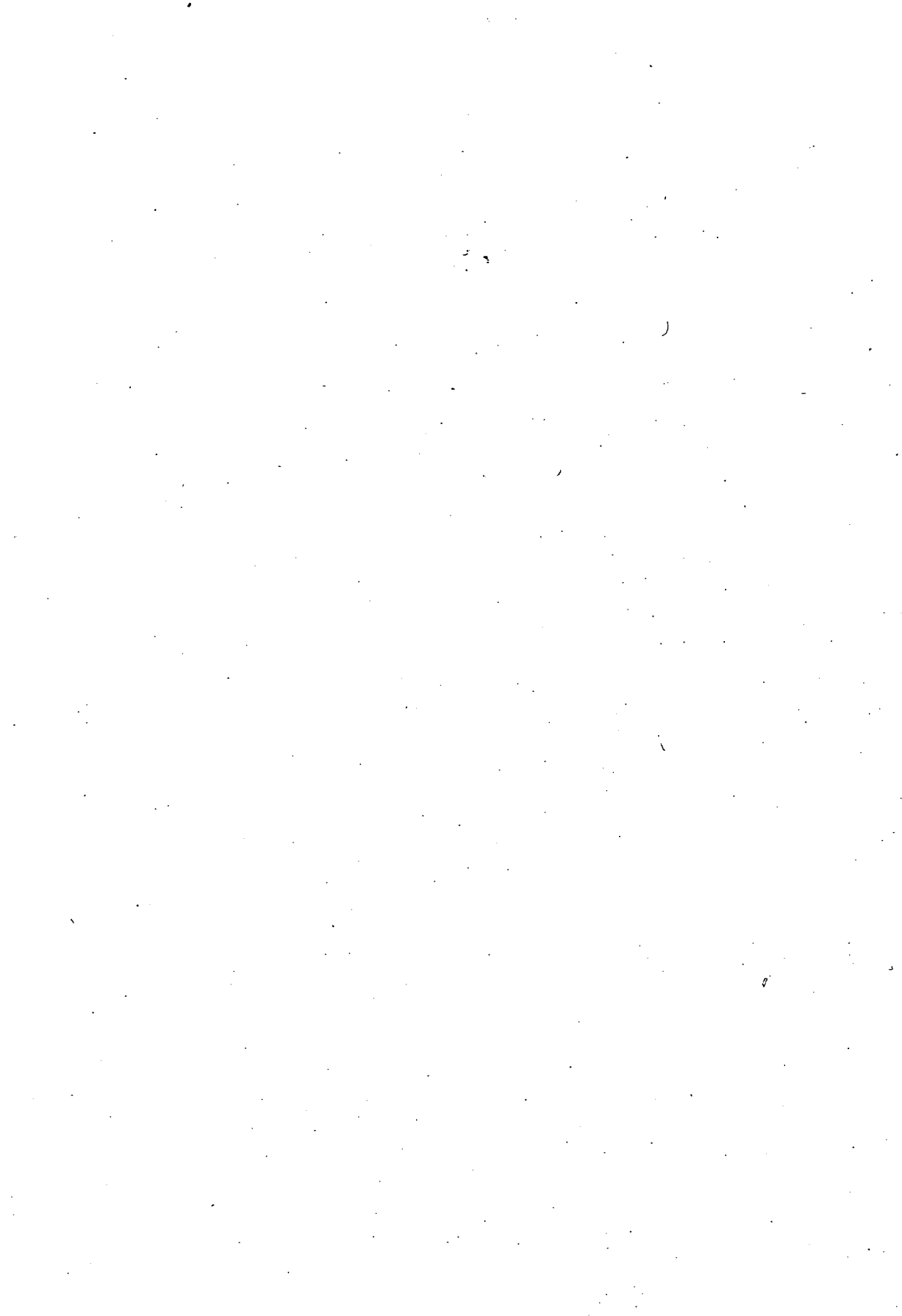
References

- [1] CODD, E. F., *Cellular automata*, Academic Press, Inc. New York, London, 1968.
- [2] VOLLMAR, R., Über einen Interpreter zur Simulation Zellularen Automaten, *Angewandte Informatik*, v. 6, 1973, pp. 249—256.
- [3] BRENDER, R. F., *A programming system for the simulation of cellular spaces*, Ph. D. Thesis, The University of Michigan, Ann Arbor, 1970.
- [4] BAKER, R., G. T. HERMAN, CELIA — a cellular linear iterative array simulator, *Proceedings of the Fourth Conference on Applications of Simulation*, 1970, pp. 64—73.
- [5] WU-HUNG LIU, CELIA, *Users manual*, Dept. of Computer Science, State University of New York at Buffalo, October, 1972.
- [6] *Cellular spaces, homogeneous structures*, Institute of Mathematical Machines, Warsaw, 1973 (in Russian).
- [7] LEGENDI, T., Simulation and synthesis of cellular automata, *Conference on Programming Systems'75*, Szeged, 1975, pp. 210—217 (in Hungarian).
- [8] LEGENDI, T., Simulation of cellular automata, the simulation language CELLAS, *Conference on Simulation in medical, technical and economy sciences*, Pécs, 1975, pp. 100—106 (in Hungarian).
- [9] CZIBIK, I., T. LEGENDI, User's manual CELLAS 1.0, 1976 (in Hungarian).
- [10] LEGENDI, T., GY. HEGEDŰS, L. PÁLVÖLGYI, User's manual INTERCELLAS, 1976 (in Hungarian).
- [11] LEGENDI, T., Cellprocessors in computer architecture, *Computational Linguistics and Computer Languages XI*, 1976, pp. 147—167.
- [12] LEGENDI, T., TRANSCCELL — a cellular space transition function definition and minimization language, to appear in *Computational Linguistics and Computer Languages XII*.

(Received Oct. 13, 1976)









INDEX — TARTALOM

<i>A. Benczúr</i> and <i>A. Krápli</i> : A note on data base integrity	181
<i>A. Ádám</i> : On some open problems of applied automaton theory graph theory (suggested by the mathematical modelling of certain neuronal networks).....	187
<i>I. Pávó</i> : Topological analysis of linear systems	215
<i>L. T. Kóczy</i> : On some basic theoretical problems of fuzzy mathematics	225
<i>I. Fekete</i> and <i>L. Varga</i> : On the formal definition of VDL-objects	239
<i>P. Ecsedi-Tóth</i> and <i>A. Varga</i> : An effective theorem proving algorithm	249
<i>T. Legendi</i> : INTERCELLAS an interactive cellular space simulation language	261

ISSN 0324—721 X

Felelős szerkesztő és kiadó: Gécseg Ferenc
A kézirat a nyomdába érkezett: 1977. július hó
Megjelenés: 1977. december hó
Példányszám: 1000. Terjedelem: 7,87 (A/5) ív
Készült monószedéssel, íves magasnyomással
az MSZ 5601 és az MSZ 5602—55 szabvány szerint
77-3067—Szegedi Nyomda—F. v.: Dobó József