ISSN 2782-2427

CONTROL SCIENCES

4/2021

[1]Russian Academy of Sciences.

CONTROL SCIENCES
Scientific Technical Journal

6 issues per year
ISSN 2782-2427
Open access

Published since 2021

Original Russian Edition
Problemy Upravleniya
Published since 2003

FOUNDER AND PUBLISHER
V.A. Trapeznikov
Institute of Control Sciences
of Russian Academy of Sciences

Editor-in-Chief
D.A. Novikov, Corr. Member of RAS

Deputy Editor-in-Chief
F.F. Pashchenko

Executive Editor-in-Chief
N.E. Maximova

Editor
L.V. Petrakova

Editorial address
65 Profsoyuznaya st., office 410,
Moscow 117997, Russia

☎/🖹 +7(495) 198-17-20, ext. 1410

✉ pu@ipu.ru

URL: http://controlsciences.org

Published: August 30, 2021

Registration certificate of
Эл № ФС 77-80482
of 17 February 2021
issued by the Federal Service
for Supervision of Communications,
Information Technology, and Mass
Media

CONTROL SCIENCES

4.2021

CONTENTS

DOI: http://doi.org/10.25728/cs.2021.4.1

NATURAL COMPUTING WITH APPLICATION TO RISK MANAGEMENT IN COMPLEX SYSTEMS

A.A. Shiroky and A.O. Kalashnikov

Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia

✉ shiroky@ipu.ru, ✉ aokalash@ipu.ru

Abstract. This paper surveys natural computing models and methods with application to risk management in complex systems. The equivalence of risk minimization and effective control problems is shown. The general risk management problem is stated for complex systems under uncertainty. The structure of fundamental and applied risk management problems is described. The well-known natural computing methods are briefly considered with application to risk management by the criteria of formalism, universality, and learning capability. The scientific community's preferences in natural computing models and methods for solving different classes of risk management problems are analyzed. Some promising approaches are outlined, which are currently underinvestigated according to the authors' opinion.

Keywords: risk management, effective control problem, natural computing.

INTRODUCTION

Many control problems for complex systems can be reduced to assessing and minimizing risks of the controlled system or interpreted in these terms. Risk management is an integral problem involving subproblems, including identification, monitoring, forecasting, optimization, etc. Effective solutions are impossible under continuously circulating volumetric data flows without building sufficiently complex models of controlled systems and applying adequate computational methods. The real dynamic systems for which such problems are considered are open. Hence, the methods used should adapt the models to a continuously changing (particularly unpredictable) environment.

In this regard, an analogy with the processes in living nature suggests itself: living organisms permanently struggle for survival in a variable environment causing both predictable and unpredictable threats. Analysis of the behavioral models of living organisms, group control structures, and protection mechanisms against external threats seems to be very promising due to the proven operability of such models (there are many examples in nature) and their high effectiveness achieved in the course of evolutionary processes. At the moment, natural computing is an actively developing field of science that combines element connectivity, group behavior, and emergence. In the book [1], the term "*natural computing*" was defined as an area of research that studies models and computational methods inspired by nature and considers the phenomena occurring in nature from the information processing viewpoint.

Many branches of natural computing (e.g., artificial neural networks) are now at their peak of popularity; others (e.g., bacterial and DNA computing) are relatively little investigated. This paper surveys the main known classes of natural computing algorithms and their application to risk management. Note that describing the approaches themselves is not the goal of the authors.

Further presentation is organized as follows. In Section 1, the general risk management problem is stated for complex systems under uncertainty, and its equivalence to the effective control problem is shown. Section 2 considers fundamental and applied risk management problems for complex systems, identifying those solved by natural computing methods. In Section 3, we suggest possible classification bases of natural computing, highlighting the ones important for applying the described algorithms and methods to risk

management in complex systems. Section 4 overviews well-known natural computing methods and algorithms and classifies them by the bases selected in Section 3. Finally, Section 5 outlines some promising approaches currently underinvestigated according to the authors' opinion.

1. RISK MANAGEMENT IN COMPLEX SYSTEMS UNDER UNCERTAINTY: GENERAL PROBLEM STATEMENT

In this paper, complex systems are systems with an infinite variety of responses to external actions. In the most general statement, the control problem is to find control actions transferring a complex system to a target state. Such control actions are called *effective*. Formally, this problem can be written as follows.

Let U be the set of control actions, and let Q and Θ be the sets of system states and environment's states, respectively. Assume that the system state $q \in Q$ is uniquely determined by the environment's state and control actions applied: $q = q(u, \theta)$, where $u \in U$ and $\theta \in \Theta$. We introduce a functional $K = K(u, q, \theta) = K(q(u, \theta))$, further referred to as *the optimality criterion*, which depends on the control action u, the system state q, and the environment's state θ. The problem is finding a *globally optimal* control action $u^* \in U$ such that

$$K(u^*, q, \theta) = \max_{u \in U} K(u, q, \theta). \quad (1)$$

In the degenerate case, when the controller "knows" the actual states of the controlled system $q_0 \in Q$ and environment $\theta_0 \in \Theta$, the optimality criterion is a function of one variable, and problem (1) reduces to:

$$K(u^*, q_0, \theta_0) = \max_{u \in U} K(u, q_0, \theta_0). \quad (2)$$

Suppose that there exists a set of (nonunique) variables (u^*, q^*, θ^*) for which

$$K(u^*, q^*, \theta^*) = \max_{u \in U} \max_{q \in Q} \max_{\theta \in \Theta} K(u, q, \theta)$$

(*the perfect statement*).

The function

$$\varphi(u, q, \theta) = K(u^*, q^*, \theta^*) - K(u, q, \theta),$$

representing the difference between the maximum and current values of the optimality criterion (for some u, q, and θ), will be called *the loss function*. Clearly,

$$\min_{u \in U} \varphi(u, q, \theta) = K(u^*, q^*, \theta^*) - \max_{u \in U} K(u, q, \theta).$$

In other words, maximization of the optimality criterion is equivalent to minimization of the loss function. In the deterministic case, the equivalence holds as well:

$$\min_{u \in U} \varphi(u, q_0, \theta_0) = K(u^*, q^*, \theta^*) - \max_{u \in U} K(u, q_0, \theta_0). \quad (3)$$

In real problem statements, complete awareness is rare, mainly occurring in technical systems control. As a rule, the controller has incomplete and (or) unreliable information about the states of the controlled system and environment. Hence, to solve the problem, the uncertainty must be eliminated somehow. For example, it is possible to choose the most probable or least favorable states, or act differently; see the paper [2].

Let $u \in U$ be a fixed control action. We define an uncertainty elimination operator $\mathfrak{J}$: the result of applying it to the optimality criterion will depend on the chosen control action u only:

$$\mathfrak{K}(u) = \mathfrak{J}K(u, \cdot, \cdot).$$

By analogy with problem (2), we find a globally optimal control action u^* that will maximize the optimality criterion under uncertainty:

$$\mathfrak{K}(u^*) = \max_{u \in U} \mathfrak{K}(u). \quad (4)$$

We introduce a real-valued *risk function*

$$\rho(u) = \mathfrak{K}(u^*) - \mathfrak{K}(u).$$

Generally, *risk* is often defined as a system parameter and a property of a controller (particularly, a decision-maker) to decide under uncertainty, which can cause both undesirable (dangerous) and significantly beneficial consequences. (For example, see the paper [3].) Within the problem statement under study, we introduce the absolute maximum of the optimality criterion. Therefore, there can be no beneficial consequences, and the values $\rho(u)$ are interpreted as an undesirable risk due to using the control action u. By analogy with (3), we pose the risk minimization problem:

$$\rho(u^*) = \min_{u \in U} \rho(u). \quad (5)$$

Obviously,

$$u^* = \operatorname{Argmax}_{u \in U} \mathfrak{K}(u) = \operatorname{Argmin}_{u \in U} \rho(u). \quad (6)$$

Thus, in the case of complex systems control under uncertainty, **maximization of the optimality criterion (4)** is equivalent to **minimization of the risk (5): the identity (6) holds**.

2. THE STRUCTURE OF RISK MANAGEMENT PROBLEMS FOR COMPLEX SYSTEMS

Without loss of generality, the risk management problem for complex systems (see the statement in Section 1) can be decomposed into several particular, sequentially arising problems.

First, to determine the components of the system under study and its operating environment, it is necessary to solve *the identification problem*. Then, for the identified objects, it is necessary to determine a refer-

ence model of their behavior, i.e., solve *the modeling problem*. Next, it is necessary to compare the behavior of these objects with the reference model *to detect anomalies* in the latter. Finally, it is required *to predict* the situation considering the anomalies detected in the previous step.

A general solution of the risk minimization problem is based on solving the four problems mentioned. Note that these problems do not depend on the essence and structure of the system under study. At the same time, to build a technology for their solution, it is necessary to consider the subject area specifics, passing to applied and technological risk management problems. There are four classes of such problems; see Fig. 1.

Let us correlate the fundamental and applied risk management problems for complex systems (Fig. 1). When solving the fundamental problem of objects identification, we determine their most essential parameters. The set (e.g., Cartesian product) of the ranges of all objects parameters in the model is the state space of the system and its environment. Such a space is constructed by solving the applied parameterization problem.

Different objects of the system and its environment can exhibit different behavior. Therefore, to start solving the fundamental problem of behavior modeling, it is first necessary to solve the applied problem of classifying objects of the controlled system.

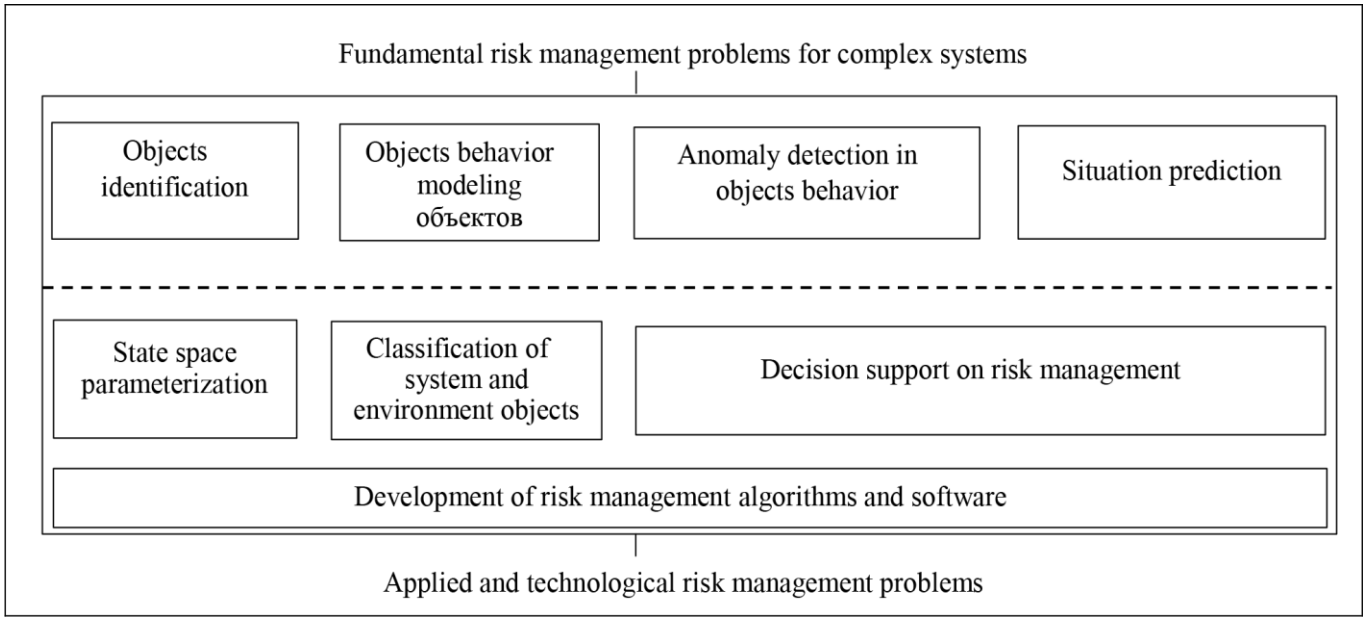


Fig. 1. The hierarchical structure of risk management problems for complex systems.

The fundamental problems of identifying behavioral anomalies and predicting the situation are associated at the application level with algorithms for classification, behavior modeling, anomaly detection, and prediction. In risk management, they are used together. At the same time, the development of decision-making algorithms is a separate applied problem that depends on the subject area but, as a rule, not on the implementation of other algorithms.

The natural computing algorithms and methods considered below are intended for solving the entire spectrum of the applied problems of classification, behavior modeling, anomaly detection, and prediction. From the authors' viewpoint, a good method should suit any of these problems, including the development of algorithms for application software. Further considerations will rest on this premise.

3. CLASSIFICATION OF NATURAL COMPUTING MODELS AND METHODS

When classifying natural computing, researchers often select as a basis the *biological process* (biological evolution, brain activity, sensory organs activity, etc.) underlying the algorithm or model of computations. This basis is convenient since it separates well the classes of algorithms. Its disadvantage lies in not answering the question: which problems can be solved by different classes of algorithms?

Another approach to classifying natural computing models and methods is selecting *the principle of algorithm development* as a basis. In the survey [4], all natural computing algorithms were divided into evolutionary, swarm, and ecological ones. Note that these

classes overlap: for example, artificial immune networks have characteristic features inherent in both swarm and evolutionary algorithms.

When considering natural computing models and algorithms with application to risk management in complex systems, it seems reasonable to select the classification bases corresponding to the specifics of problem (6). From the authors' viewpoint, the specifics are the variety of responses of such systems to external actions, not described by a finite set. Under uncertainty, the set of external actions Θ is not fully identified. Standard optimization techniques will only work in the identified part of this set. At the same time, a very important task is to minimize the risk in the event of new (previously unpredictable) external actions.

Considered within a single paradigm, the technological problem of developing algorithms and software requires a *universal* approach. In other words, the chosen approach should solve all applied risk management problems for complex systems. Therefore, the first (and most important) classification basis for natural computing models and methods will be universality. Since the matter concerns computational models, universality is understood primarily as the ability to implement an arbitrary algorithm (*universal computer*) or, at least, to calculate an arbitrary function with a given accuracy (*universal approximator*).

The risk management problem statement for complex systems assumes that the method or algorithm used must have a *formalism* adequate to the problem. In addition, since the systems under study often operate in a variable environment, the form of functions can (and will) change over time. This fact implies another requirement to the algorithm or method used, known as *learning capability*.

Thus, to solve the problem, it is advisable to adopt the following classification bases:

– universality (universal computer or universal approximator),

– the type of calculations (formal grammar, mathematical model, family of algorithms, element base, etc.),

– formalism (there is a formal model or not),

– learning capability (the ability to implement machine learning algorithms, the ability to implement adaptive algorithms, or no learning capability),

– technical implementation (hardware and software platforms).

4. NATURAL COMPUTING: A SURVEY

Natural computing models and methods include heuristic algorithms (or their families united by a common idea), some formal grammars, element bases of computers, and a series of mathematical models. The former include artificial immune systems, swarm intelligence, and amorphous and evolutionary computing. Formal grammars describe membrane computing [5] and Lindenmayer systems [6] (P systems and L-systems, respectively). The natural element bases of computers are founded on DNA molecules [7], Physarum amoebas [8], and some types of bacteria [9, 10]. Natural computing based on mathematical models includes cellular automata [11], artificial neural networks [12], and calculations with dynamic systems and chaos [13]. Let us briefly describe some of them.

4.1. Formal grammars

A formal **P system** is defined by an aggregate

$\Pi = (O, C, \mu, w_1, \ldots, w_m, R_1, \ldots, R_m, i_o)$,

where:

O is a non-empty finite alphabet of *objects*;

$C \subset O$ is the set of *catalysts*;

μ is a membrane structure composed of membranes $1, \ldots, m$ that determine the *regions* of the P system;

$w_1, \ldots, w_m$ are strings over the alphabet O that describe *the multisets of objects* contained in regions $1, \ldots, m$, respectively;

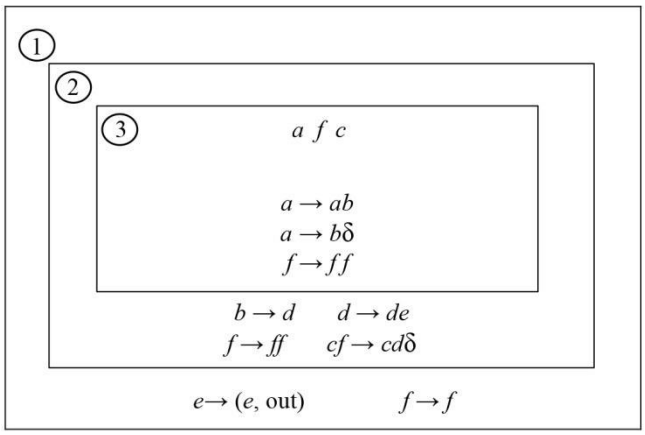
$R_1, \ldots, R_m$ is a finite *set of evolution rules* for regions $1, \ldots, m$, respectively;

finally, $i_o \in \{0, \cdots, m\}$ is the number of the region containing the result of calculations (zero result will be sent to the environment).

The evolution rules have the form $u \to v$ or $u \to v\delta$, where $u \in O^+$, $v \in (O \times Tar)^*$, and $Tar = \{here, in, out\}$. Here, O^+ denotes the set of all possible strings over the alphabet O, except the empty string λ, and $O^* = O^+ + \{\lambda\}$. Rules with an arbitrary action u are called *cooperative*; rules with $u \in (O - C)$, *noncooperative*. Noncooperative rules with a catalyst, i.e., the ones of the form $cu \to cv$ or $cu \to cv\delta$, where $u \in (O - C)$, $c \in C$, and $v \in ((O - C) \times Tar)^*$, are called *catalytic*. The catalysts in such rules "help" other objects of the P system to evolve, not changing themselves and never moving between its regions.

The membrane structure and multisets of objects located in the regions bounded by the membranes define the configuration of the P system. The initial configuration is given by the membrane structure and the objects contained in the regions $(\mu, w_1, \ldots, w_m)$. In the course of the system evolution, applying the rules can change both the multisets of objects and the membrane structure. An example of the initial configuration of a P system is shown in Fig. 2.

Note that the rules of evolution in each step are applied in all regions of the system simultaneously.



Fif. 2. Initial configuration of P system with three membranes, alphabet $O = \{a, b, c, d, e, f, \delta\}$, and the set of catalysts $C = \{c\}$.

A detailed description of membrane computers can be found in [5].

A basic **L-system** consists of an alphabet V, an initial string (*axiom*) $\omega \in V^+$, where V is the set of all nonzero-length strings over V, and the set of production rules P of the form p: $a \to x$, $a \in V$, $x \in V^+$. For each symbol $a \in V$ not belonging to the left-hand side of P, the rule $a \to a$ holds. Such symbols are called *constants* or *terminal symbols*.

The rules from the set P are applied iteratively, starting from the axiom (initial state). At each iteration, all symbols from the left-hand side of P are replaced by the corresponding right-hand side. For example, the system $G = (V = \{a, b\}, \omega = a, P = \{a \to b, b \to ba\})$ produces the strings $a \to b \to ba \to ba\ b \to ba\ b\ ba \to ba\ b\ ba\ ba\ b \to \ldots$

Basic L-systems (called 0L-systems) involve no special symbols. More complex symbolic L-systems (SL-systems) recognize special substrings, i.e., reserved symbol operators. A detailed description of L-systems and particularly these operators, including several practical examples, can be found in [6].

4.2. Element bases

DNA computers encode information in nucleotide sequences. DNA molecules contain four nitrogenous bases: cytosine (C), guanine (G), adenine (A), and thymine (T). In the DNA computer model, nucleotide sequences are represented as strings over a finite alphabet $\{A, T, G, C\}$. The operations performed by the computer using various enzymes on DNA molecules are completely described by the rules of extended H-systems (also called splicing systems, see Chapter 7 of the book [14]).

Let

$$S = (V, \Sigma, A, R),$$

where: V is a finite alphabet; $\Sigma \in V$ is a terminal alphabet; $A \subset V^*$ is the set of axioms; finally, $A \subset V^* \times V^* \times V^* \times V^*$ are splicing rules.

Each rule $r\ r = (u_1, u_2, u_3, u_4) \in R$ is written as

$$r = u_1 \# u_2 \$ u_3 \# u_4;\ u_1, u_2, u_3, u_4 \in V^*;\ \#, \$ \notin V.$$

This expression is interpreted as follows: cut the string between consecutive entries of the substrings u_1 and u_2, u_3 and u_4; then glue the resulting fragments on the left of the first cut and on the right of the second one so that u_1 and u_4 turn out near each other. For example, consider some $x_1, x_2, y_1, y_2 \in V^*$ and strings $x = x_1 u_1 u_2 x_2$ and $y = y_1 u_3 u_4 y_2$. Applying the rule r to the pair (x, y) yields the string $z = x_1 u_1 u_4 y_2$.

The formal model of a DNA computer was described in detail in [15]. Bacterial computing [9, 10] is an implementation of such a computer in living organisms.

4.3. Mathematical models

A cellular automaton is defined by an aggregate $(\mathcal{L}, \mathcal{S}, \mathcal{N}, f)$, where:

$\mathcal{L} \subseteq \mathbb{Z}^D$ is the D-dimensional cellular space, i.e., a (possibly infinite) set of cells in $\mathcal{L}$ that forms a regular lattice;

$\mathcal{S}$ is the finite set of cells states;

$\mathcal{N} = (\vec{v_1}, \ldots, \vec{v_N})$ is the neighborhood vector from N elements of the set $\mathcal{L}$ that connects a cell with its neighbors (for a cell located in $\vec{v} \in \mathcal{L}$, the neighbor cells are located in $(\vec{v} + \vec{v_i}) \in \mathcal{L}\ \forall i \in \{1, \ldots, N\}$);

Finally, $f: S^N \to S$ is a local transition rule determining the cell state at the next time instant, $f(a_1, \ldots, a_N)$, where $a_1, \ldots, a_N$ are the states of its N neighbors.

A *configuration* of a cellular automaton is a global state $C: \mathcal{L} \to \mathcal{S}$. Let $\mathcal{S}^{\mathcal{L}}$ be the set of all configurations. Then a mapping $G: \mathcal{S}^{\mathcal{L}} \to \mathcal{S}^{\mathcal{L}}$ is called a global transition function, and a cellular automaton can be defined by the aggregate $\left(\mathcal{L},\ \mathcal{S},\ \mathcal{S}^{\mathcal{L}}, G\right)$.

Computing based on a cellular automaton is performed by sequentially calculating the transition function. As a rule, a cellular automaton is assumed to be synchronous (all cells change their state simultaneously) and homogeneous (all cells use a single transition rule). In this case, if a configuration $y = (y_{\vec{v}})_{\vec{v} \in v\mathcal{L}}$ follows a configuration $x = (x_{\vec{v}})_{\vec{v} \in v\mathcal{L}}$ (that is, $y = G(x)$), y will be the result of calculating the following expression for each $\vec{v} \in \mathcal{L}$:

$$y_{\vec{v}} = G(x)_{\vec{v}} = G(x_{\vec{v}}) = f\left(x_{\vec{v}+\vec{v_1}}, \ldots, x_{\vec{v}+\vec{v_N}}\right).$$

The model, types, and properties of cellular automata were considered in detail in the book [16].

The formal model of **an artificial neuron** is written as

$$s = \mathbf{w} \cdot \mathbf{x} + b,\ z = g(s),$$

where: $\mathbf{x} \in \mathbb{Z}_{\{0,1\}}^{N}$ denotes the vector of *input data*; $\mathbf{w}$ denotes the vector of *weights*; b is the displacement; $g(\cdot)$ is *an activation function*; finally, z denotes *the output*. When forming an artificial neural network, individual neurons s are combined into a layer

$$\mathbf{s} = \mathbf{W}\mathbf{x} + \mathbf{b},\ \mathbf{z} = g(\mathbf{s}),$$

where $\mathbf{W}$ denotes the matrix of weights. A neural network can have several interconnected layers.

An artificial neural network is trained by adjusting the weights, as a rule, via minimizing some loss functional: $L(\mathbf{w}) \to \min_{\mathbf{w}}$.

The evolution of artificial neural network architectures, learning modes, and applications was discussed in the survey [17].

The computing model with dynamic systems and chaos is based on the concept of a chaotic element: *a chaotic chip* or *a chaotic processor* [18]. Let the state of such an element be determined by the variable x. The logical gate implemented by this element is constructed in the following way.

- The input signals are:

$x \to x_0 + X_1 + X_2$ for the binary logical operations (NAND, NOR, XOR, AND, OR, and XNOR);

$x \to x_0 + X$ for the unary operations (NOT),

where x_0 denotes the initial system state and

$$X = \begin{cases} 0 & \text{for } I = 0, \\ V_{in} > 0 & \text{for } I = 1, \end{cases}$$

where V_{in} is a positive constant.

- Dynamic updating is used, i.e., $x \to f(x)$, where $f(x)$ is a linear function.
- A threshold mechanism is used for obtaining the output signal Z:

$$Z = \begin{cases} 0 & \text{if } f(x) \le E, \\ f(x) - E & \text{if } f(x) > E, \end{cases}$$

where E is a threshold.

The output value is interpreted as logical zero if $Z = 0$ and as logical one if $Z > 0$.

A feature of chaotic elements is the ability to change the type of the logical gate they implement using control of the values (x_0, E). A classical example is a chaotic element with $f(x) = 4x(1 - x)$ and V_{in}=1/4. This element implements one of the logical operations AND, OR, XOR, NAND, or NOT, depending on the values (x_0, E); see Table 1.

The chaotic computing paradigm was considered in detail in the paper [18].

Table 1

Values x_0 and E corresponding to different logical operations of chaotic element with $f(x) = 4x(1 - x)$ and V_{in}=1/4

Logical operation	AND	OR	XOR	NAND	NOT
x_0	0	1/8	¼	3/8	1/2
E	3/4	11/16	¾	11/16	3/4

The other approaches are either unique in natural computing or combine very heterogeneous elements and therefore cannot be identified with any of the groups mentioned.

4.4. Universality of natural computing

Consider the universality of different types of natural computing.

Cellular automata and artificial neural networks are universal computers. Smith's classical work [19] was devoted to the Turing-complete cellular automaton. The Turing completeness theorem for fully connected recurrent artificial neural networks with sigmoidal activation functions was proved in the paper [20].

The computing paradigm with nonlinear dynamic systems (chaos) emerged in response to the limitations of transistors as a traditional element base. Logical elements (gates) based on transistors cannot be changed after the hardware implementation. Programming involves switching between several different single-purpose elements. A chaotic element is capable of transforming into various logical elements using a threshold-based morphing mechanism. The work [21] described the implementation of the AND, OR, NOT, and XOR gates, substantiating the universality of chaotic chips.

Collision-based computing [22], also called ballistic computing, free space computing, and billiard computing in the literature, implement logical circuits using a homogeneous unstructured environment with mobile localizations. These can be gliders in cellular automata, solitons in optical systems, or wave fragments in excitable chemical systems. Logical truth corresponds to the presence of localization; logical false, to the absence of localization. When two or more moving localizations collide, they change their velocity and (or) state vectors. The post-collision paths and (or) states of localizations result from logical operations implemented by a collision. The equivalence of the billiard computing model based on a two-dimensional block cellular automaton to a reversible Turing machine was established in [23].

The term "reaction-diffusion computing" can be interpreted as a computational model based on the corresponding family of differential equations and as a chemical computer in which data are represented by concentrations of different chemical elements and processing is performed using chemical reactions (e.g., the Belousov–Zhabotinsky reaction); see [24, 25]. On the other hand, reaction-diffusion equations are often used to model other computational environments, particularly chaos and collision-based computing. The paper [26] demonstrated the fundamental possibility of universal calculations in chemical computers under limited resources. Note that a similar result was obtained therein for Physarum computing, discussed below.

Now let us consider formal grammars. Membrane computing (also known as P systems [5]) involves the membrane concept based on the simplest analogy with biological cells. During the operation of such systems, the objects of calculations (the sets of symbolic multisets) and, moreover, the membrane structures themselves evolve. This peculiarity is inherent in P systems. Each membrane structure (cell) is treated as a separate computational element, and the principle of maximum parallelism is employed. Consequently, such a grammar allows writing P-systems that solve NP-complete problems in polynomial time (due to the exponential growth of the number of membranes and, accordingly, parallelism). Computational completeness was proved for some P systems. In particular, this was done for communicative membrane systems [27]. (In such systems, actions over an object are performed when it passes through the membrane.) In the paper [28], a similar assertion was derived for several classes of P systems with catalysts (symbols in a multiset that cause the application of some rule, not "consumed" by it). In 2009, Turing completeness was shown for an extension of P systems, the so-called mutual mobile membrane systems, in the case of three membranes [29].

Lindenmayer systems (L-systems) were originally proposed by botanist A. Lindenmeier [30] as a formal language for describing algal growth. Later on, the approach was developed into a formal grammar. In 1991, L-systems were first used for expanding some elementary functions into a Taylor series [6]. A corresponding compiler and a visualization subsystem for the IBM PC platform were created soon [31]. However, the generation of fractal structures remains the main application of Lindenmayer systems. The computational completeness of such systems is not discussed in the literature.

A separate trend in the development of natural computing is searching for new (alternative) computational element bases. In 1994, it was proposed to use a DNA molecule [7]. An advantage of the DNA approach is the simultaneous generation of all possible solutions of combinatorial problems (e.g., Hamiltonian path problems in directed graphs) using known biochemical reactions. Then the molecule string encoding the desired response can be quickly filtered out. However, when scaling the method proposed by L. Adleman, the number of DNA molecules needed to find a solution grows exponentially with the problem dimensionality. This fact imposes physical restrictions on the computational power of such a computer. Nevertheless, already in 1999, it was established that DNA computers can be universal computers [32]. In 2017, a design of a DNA computer implementing a nondeterministic universal Turing machine was proposed [33].

In 2010, using Adleman's ideas, a group of researchers created a bacterial computer based on genetically modified E. coli bacteria [10]. In contrast to the earlier experiments [9], the researchers placed the DNA sequences encoding the problem to the breaks in the DNA strands of genes encoding fluorescent proteins. Note that the inserted fragments were framed by the so-called hix sites. As a result, the effect of site-specific inversion of bacterial DNA in the presence of a special protein, DNA-invertase Hin, was used. During the inversion of the nucleotide strand, the encoding fluorescent proteins are restored, and the bacteria that "found" the correct solution glow under the microscope.

Another approach to the creation of non-traditional computers is to use some special properties of living organisms. The most famous experiment involves the slime mould (amoeba) Physarum polycephalum, which tends to take a shape minimizing the sunlight effect on it. The paper [34] described the application of such an "amoebic computer" to the approximate solution of the NP-complete traveling salesman problem. As demonstrated therein, the amoeba solves this problem in linear time. However, to calculate the lighting scheme of the amoeba, a recurrent neural network is used, whose dynamics are determined by a weight matrix with n^4 elements (n denotes the number of cities). Therefore, the gain in time is not obvious for large dimensions. The universality of such a computer has not yet been investigated.

Finally, we consider the families of heuristic and metaheuristic algorithms commonly related to natural computing. Artificial immune systems include several classes of such algorithms, created by analogy with their natural implementation in the immune systems of vertebrates. Despite no generally accepted formalization (the one proposed in [35] has not become wide-

spread), the results of [36, 37] suggest that a universal approximator can be based on an artificial immune system.

Swarm intelligence algorithms usually include system models in which many agents interact locally with the environment and with each other. The agents obey some fairly simple rules of behavior. However, the entire multi-agent system exhibits complex ("intelligent") behavior. In practice, such algorithms are used for solving various optimization problems. For example, see the survey [38]. The universality of such algorithms (e.g., as a universal approximator) has apparently not been raised in the literature.

The term "amorphous computing" was introduced by a group of MIT researchers in 1996. It refers to a class of computing devices consisting of numerous cheap, almost identical information processing units. In this case, cheapness is an essential property: among ther things, such devices were intended for the wholesale production of smart structural materials as an additive ("improver"). The possibility of universal calculations on an amorphous computer composed of asynchronously operating finite probabilistic automata was first shown in [39]. Several amorphous computing systems from the class of universal computers were considered in [40].

Along with the fuzzy logic and swarm intelligence models considered previously, evolutionary computing belongs to a large class of the so-called "soft computing" with approximate models. Technically, evolutionary computing is a family of global optimization algorithms based on the idea of biological evolution. The family of candidate solutions forms a "population" that is gradually improved by selection or random "mutations." The process stops when the solutions reach the required level of accuracy.

The forms of natural computing, the types of corresponding calculations, and their universality are described in Table 2, including the relevant references.

5. NATURAL COMPUTING FOR RISK MANAGEMENT IN COMPLEX SYSTEMS: ANALYSIS OF POSSIBLE USE

As noted earlier, natural computing tools suitable for risk management in complex systems must, first of all, be universal, have a formal mathematical model, and, finally, demonstrate a high level of adaptiveness. In terms of applications, it is also important that the model or algorithm have software implementation (to be used when developing specialized software) or hardware implementation (to speed up operation). The latter requirement becomes crucial for risk management in information security (e.g., when creating traffic analyzers). In the previous section, we have selected the types of natural computing with proven universality. Let us check whether they match the other criteria.

Table 2

Well-known natural computing methods and algorithms and their universality

Form of natural computing	Type of calculations	Universality
Cellular automata	Mathematical model, element base	Universal computer [19]
Artificial neural networks	Mathematical model, element base	Universal computer [20]
Computing with dynamic systems and chaos	Mathematical model, element base	Universal computer [21]
Collision-based computing	Computational model	Universal computer [23]
Reaction-diffusion computing	Mathematical model, element base	Universal computer [26]
Membrane computing (P systems)	Formal grammar	Universal computer [27–29]
Lindenmayer systems (L-systems)	Formal grammar	Applicable to symbolic calculations [6]
DNA computing	Element base	Universal computer [32, 33]
Bacterial computing	Element base	Universal computer (possibly)
Physarum computing	Element base	Universal computer [26]
Artificial immune systems	Family of algorithms	Universal approximator (possibly) [36, 37]
Swarm intelligence	Family of algorithms	Universal approximator (possibly)
Amorphous computing	Family of algorithms	Universal computer [39, 40]
Evolutionary computing	Family of metaheuristic algorithms	Not applicable

According to the Curtis–Hedlund–Lyndon theorem for cellular automata, proved in 1969, transitions between any two shift spaces can be determined by a uniformly local rule [41]. The class of adaptive stochastic cellular automata allows implementing adaptive algorithms [42]. Technically, the formal automaton model is implemented on a conventional computer: free software modules are available in Python and Wolfram (Mathematica). Thus, adaptive control systems can be designed based on this type of computing.

Artificial neural networks received a formal model within finite automata theory back in 1956; see [43]. One year later, the classical work was published describing the learning algorithm for an artificial neural network based on a two-layer perceptron in which one layer is hidden and untrainable [44]. Many programmed neural network architectures are currently available for solving various classes of problems (mainly recognition and classification). Their hardware implementations are constantly being improved; for example, see the survey [45].

Chaotic calculations involve models of various nonlinear dynamic systems [13], which give the necessary formalism. The behavior of such systems can be numerically simulated, e.g., in MATLAB. Work is now underway towards the design of a chaotic processor. In particular, the paper [18] described the operation of chaotic transistors. The same research team patented the architecture of an arithmetic-logic device based on such transistors [46]. However, this class of systems does not exhibit learning capability.

Collision-based computing systems are often called billiard computers, and their model is often called the formal billiard model. The paper [47] described all currently used mobile and stationary localizations ("balls" and "tables," respectively). Such systems have no learning capability. The models of such computers can be constructed in MATLAB. There is no hardware implementation yet; the most recent known publication in this area [48] presented the idea of a chemical transistor using chemical wave fragments as stationary localizations.

The Kolmogorov – Petrovsky – Piskunov equation is considered the first formal reaction-diffusion system model in the one-dimensional case [49]. More complex models are required to describe environments suitable for computing. One example is a chemical computer based on the Belousov – Zhabotinsky reaction [26, 50]. Such systems are not trainable. ReaDDy 2 library [51] is the contemporary software implementation of the "reaction-diffusion" computing model. Also, some experiments were carried out with the programming of a chemical computer [24].

The main component of a membrane computer (membrane structure) was formally described by Paun, the pioneer of P systems [5]. Such a grammar allows writing adaptive algorithms. However, the membrane computer has no learning capability. Its software implementation is the P-Lingua programming language [52], a plug-in to Eclipse IDE. There are no hardware implementations.

Natural computing systems with alternative bases (DNA, bacterial, and Physarum computers) have no unique computational model. They use classical ones instead: a Turing machine, nondeterministic finite automata, or others, depending on the researcher's preferences. Hence, such computers are without "innate" learning capability, although they allow implementing adaptive and even self-learning algorithms. All such computers have hardware implementation; see the references in Table 3.

Most researchers still consider artificial immune systems as a set of heuristic algorithms. In 1998, an attempt was undertaken to base such systems on the formal peptide model [35], but this formalism is not yet generally accepted. Perhaps this circumstance hinders the development of software modules for implementing artificial immune systems-based solutions: several software products found by the authors (see Table 3) are not being developed at the moment. Immune networks have the "innate" learning capability and are comparable by the potential to artificial neural networks; for example, see [53].

Swarm intelligence algorithms do not rest on a single formal model of a swarm's element (particle, agent, etc.). They are united by the principle of local interaction of elements with each other and the environment. Accordingly, such algorithms cannot be trained, although some prediction problems are solved using swarm algorithms. The most popular representative of this family – the particle swarm method – has several software implementations in the form of plug-ins for MATLAB [54] and SCILAB [55]. The other swarm algorithms are less common and usually implemented by researchers to solve particular narrow problems.

Amorphous computing also does not imply an underlying formal model: it denotes a class of computing systems with a very high degree of parallelism. Specialized programming languages were developed to facilitate the programming of such systems (e.g., sensor networks). For details, see Table 3.

Table 3

Natural computing methods and algorithms and their characteristics: the presence of formal model, its learning capability, and examples of technical implementation

Form of natural computing	Formalism	Learning capability	Technical implementation
Cellular automata	Yes [41]	Adaptiveness [42]	Software platform (Python, Wolfram Mathematica)
Artificial neural networks	Yes [43]	Yes [44]	Software implementations (PyTorch, TensorFlow, etc.) and hardware implementations (see the survey [45])
Computing with dynamic systems and chaos	Yes [13]	No	Software implementation (MATLAB) and hardware implementation [18]
Collision-based computing	Yes [47]	No	Software implementation (MATLAB)
Reaction-diffusion computing	Yes [49, 50]	No	Software implementation (ReaDDy library for Python and Java [51])
Membrane computing (P systems)	Yes [5]	No	Software implementation (P-lingua language [52])
DNA computing	Yes [15]	No	DNA self-assembly [56]
Bacterial computing	No	No	Genetically modified E. Coli bacteria [9]
Physarum computing	No	No	Amoeba-based computing system [34]
Artificial immune systems	Yes [35]	Yes [53]	Software implementation (Jisys [57], iNet Framework [58] and libtissue [59] libraries)
Swarm intelligence	No	No	Software implementation for PSO [54, 55]
Amorphous computing	No	No	GPL (Growing Point Language) [60] and Proto language [61]

6. RISK MANAGEMENT WITH NATURAL COMPUTING

The formal analysis presented above identifies natural computing methods suitable for solving risk management problems in complex systems. These are cellular automata, artificial neural networks, and artificial immune systems.

To compare the prevalence of these approaches in fundamental and applied risk management problems for complex systems, we will consider the number of Google Scholar publications on the subject (search results for the requests containing the problem statement with synonymous constructs and the method name; see Tables 4 and 6). Note that the effectiveness of the methods in particular problems will be neither assessed nor compared. The number of publications rather shows the "popularity" of the approach in the scientific community, indirectly characterizing the depth of development.

The requests are structured as follows. The text of an appropriate publication should contain the problem statement (e.g., "identification" for the identification problem; "forecast OR prediction" for the prediction problem). Also, the text of an appropriate publication should contain the corresponding natural computing method or model (e.g., "artificial immune system" for artificial immune systems): its full name or generally accepted abbreviation. Synonyms in the requests are separated by the OR operator.

Note that the high results obtained for artificial neural networks could be even higher: Google Scholar limits the maximum execution time of each search request and stops viewing the publication index after it. Thus, this approach to risk management in complex systems is used much more often than the other artificial intelligence models. The conclusion concerns both fundamental and applied and technological problems (see Table 7).

Cellular automata are inferior in the frequency of use to both neural and immune networks. Rare publications involve them in identifying anomalies or as a component of an information system. Nevertheless, anomaly detection is successfully solved by cellular automata. We mention the paper [67] among the modern research on this topic.

Artificial immune networks have been developing since recently, showing themselves to be a rather promising approach. This explains their position between cellular automata and neural networks by the

Table 4

Google Scholar publications on fundamental risk management problems for complex systems

Class of problems	Search requests and examples of search results
Cellular automata	
Identification	intext: “identification” allintitle: “by cellular automata” OR “cellular automata for” – “identification of cellular automata” – “identification of optimal cellular automata” – “identification number” [62, 63]
Behavior modeling	allintext: “behavior” “activity” “modeling” OR “simulation” allintitle: “by cellular automata” OR “cellular automata for” [64, 65]
Anomaly detection	allintext: “anomaly detection” allintitle: “by cellular automata” OR “cellular automata for” [66, 67]
Prediction	allintext: “forecast” OR “prediction” allintitle: “by cellular automata” OR “cellular automata for” [68, 69]
Artificial neural networks	
Identification	identification “artificial neural network” OR “neural network” OR “deep learning” – “identification of ANN” – “identification of neural network” – “identification number” – “neural network identification” [70, 71]
Behavior modeling	behavior activity (simulation OR modelling) AND (“artificial neural network” OR “neural network” OR “deep learning”) – “neural network dynamics” – “neural network training” [72, 73]
Anomaly detection	“anomaly detection” AND (“artificial neural network” OR “neural network” OR “deep learning”) [74, 75]
Prediction	(forecast OR prediction) AND (“artificial neural network” OR “neural network” OR “deep learning”) [76, 77]
Artificial immune networks	
Identification	allintext: “identification” “computing” “artificial immune system” allintitle: “AIS” OR “immune system” OR “artificial immune” – “identification of AIS” – “identification of artificial immune system” – “identification number” – “immune system identification” [78, 79]
Behavior modeling	allintext: “behavior” “activity” “modeling” OR “simulation” “artificial immune system” allintitle: “AIS” OR “immune system” [80, 81]
Anomaly detection	allintext: “anomaly detection” “artificial immune system” allintitle: “AIS” OR “immune system” [82, 83]
Prediction	allintext: “forecast” OR “prediction” “artificial immune system” allintitle: “AIS” OR “immune system” [84, 85]

number of publications on the same classes of problems. However, their use in information systems is constrained by the absence of a generally accepted formalism and, accordingly, software implementation in the form of a software library.

Again, the search engine results for the requests (Tables 5 and 7) do not substantiate the comparative “suitability” of the model for a particular class of problems but rather describe the distribution of the scientific community’s preferences. A relatively small number of references indicates that a particular method or model is less popular. The reasons may be insufficient studies of the model and the lack of convenient software tools for its use.

Table 5

Using artificial intelligence models and methods in fundamental risk management problems for complex systems (by the number of Google Scholar publications)

Problem	Cellular automata	Artificial neural networks	Artificial immune networks
Identification	2 390	17 800	5 720
Behavior modeling	1 630	20 700	2 550
Anomaly detection	56	17 200	4 170
Prediction	4 220	18 000	7 520

Table 6

Google Scholar publications on applied and technological risk management problems for complex systems

Class of problems	Search requests
Cellular automata	
Classification	allintext: “classification of” allintitle: “by cellular automata” OR “cellular automata for” [86, 87]
Decision support	allintext: “decision support” allintitle: “by cellular automata” OR “cellular automata for” [88, 89]
Development of information and control systems	allintext: “information system development” OR “software development” “control” allintitle: [96, 97]
Artificial neural networks	
Classification	“classification of” AND (“artificial neural network” OR “neural network” OR “deep learning”) – “ANN classification” – “neural network classification” – “classification of neural” [92, 93]
Decision support	“decision support” AND (“artificial neural network” OR “neural network” OR “deep learning”) [94, 95]
Development of information and control systems	“control” AND (“software development” OR “system development”) AND (“artificial neural network” OR “neural network” OR “deep learning”) [96, 97]
Artificial immune networks	
Classification	allintext: “classification of” “artificial immune system” allintitle: “AIS” OR “immune system” – “AIS classification” – “classification of artificial immune” – “classification of immune” – “immune system classification” [98, 99]
Decision support	allintext: “decision support” “artificial immune system” allintitle: “AIS” OR “immune system” [100, 101]
Development of information and control systems	allintext: “information system development” OR “software development” “control” “artificial immune system” allintitle: “AIS” OR “immune system” [102, 103]

Table 7

Using artificial intelligence models and methods in applied and technological risk management problems for complex systems (by the number of Google Scholar publications)

Problem	Cellular automata	Artificial neural networks	Artificial immune networks
Classification	4 330	18 200	4 780
Decision support	1 050	17 800	2 210
Development of information and control systems	242	17 300	628

CONCLUSIONS

The use of analytical methods for risk management in complex systems shows predictable results. However, control performance directly depends on the structure of the controlled system. For multi-agent heterogeneous open systems, an analytical solution of risk management problems can take a long time. Moreover, the resulting solution is often effective only in a small domain of the state space.

Natural computing solutions are interesting due to high adaptiveness, a consequence of the complexity of natural systems. Nevertheless, not all of these approaches have the required level of abstractness, adaptiveness, and learning capability.

This paper has analyzed the known natural computing models and methods with application to both fun-

damental and applied and technological risk management problems in complex systems. Three models have been identified that fully meet all the criteria: artificial neural networks, cellular automata, and artificial immune networks.

Artificial neural networks have been used for risk management in complex systems for many years and have proven their effectiveness. The other two approaches are less common, but their potential cannot be considered lower. The most probable reasons hindering their development are no generally accepted formalism for artificial immune networks and no learning capability for cellular automata.

It seems promising to develop risk management methods for complex systems based on cellular automata and artificial immune systems.

REFERENCES

1. *Handbook of Natural Computing*, Rozenberg, G., Bäck, T., and Kok, J.N., Eds., Berlin: Springer Berlin Heidelberg, 2012.
2. Kalashnikov, A.O., Managing the Information Risks of Organizational Systems: A General Problem Statement, *Informatsiya i Bezopasnost'*, 2016, vol. 19, no. 1, pp. 36–45. (In Russian.)
3. Kononov, D.A., Studying the Security of Control Systems by Analyzing Their System Parameters, *Materialy 28-oi Mezhdunarodnoi konferentsii "Problemy upravleniya bezopasnost'yu slozhnykh sistem"* (Proceedings of 28th International Conference on Complex Systems Security Control), December 16, 2020, Moscow, Kalashnikov, A.O. and Kul'ba, V.V., Eds., Moscow: Trapeznikov Institute of Control Sciences, 2020, pp. 102–108. (In Russian.)
4. Nemade, M.N. and Rane, M.D., A Review on Bio-Inspired Computing Algorithms and Application, *Proceedings of National Conference on Recent Trends in Computer Science and Information Technology* (*NCRTCSIT-2016*), Nagpur, 2016, pp. 12–19.
5. Paun, G., Introduction to Membrane Computing, in *Applications of Membrane Computing*, Ciobanu, G., Paun, G., and Perez-Jimenez, M.J., Eds., Berlin–Heidelberg: Springer, 2006, pp. 1–42.
6. Goel, N.S. and Goodwin, M.D., Symbolic Computation Using L-systems, *Applied Mathematics and Computation*, 1991, vol. 42, no. 3, pp. 223–253.
7. Adleman, L.M., Molecular Computation of Solutions to Combinatorial Problems, *Science*, 1994, vol. 266, no. 5187, pp. 1021–1024.
8. Adamatzky, A., *Physarum Machines: Computers from Slime Mould*, vol. 74, Singapore: World Scientific, 2010.
9. Haynes, K., Broderick, M., Brown, A., et al., Engineering Bacteria to Solve the Burnt Pancake Problem, *Journal of Biological Engineering*, 2008, vol. 2, article no. 8.
10. Poet, J.L., Campbell, A.M., Eckdahl, T.T., and Heyer, L.J., Bacterial Computing, *XRDS: Crossroads, The ACM Magazine for Students*, 2010, vol. 17, no. 1, pp. 10–15.
11. Codd, E.F., *Cellular Automata*, New-York: Academic Press, 2014.
12. Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*, Cambridge: The MIT Press, 2016.
13. Ditto, W.L., Miliotis, A., Murali, K., and Sinha, S., The Chaos Computing Paradigm, in *Reviews of Nonlinear Dynamics and Complexity*, Schuster, H.G., Ed., Weinheim, Germany: Wiley-VCH, 2010, pp. 1–35.
14. Paun, G., Rozenberg, G., and Salomaa, A., *DNA Computing. New Computing Paradigma*, Berlin–Heidelberg: Springer, 1998.
15. Krasinski, T. and Sakowski, S., A Theoretical Model of the Shapiro Finite State Automaton Built on DNA, *Theoretical and Applied Informatics*, 2006, vol. 18, pp. 161–174.
16. Hadeler, K.-P. and Müller, J., *Cellular Automata: Analysis and Applications*, Cham: Springer, 2017.
17. Makarenko, A.V., Deep Neural Networks: Origins, Development, Current Status, *Control Sciences*, 2020, no. 2, pp. 3–19. (In Russian.)
18. Ditto, W.L., Miliotis, A., Murali, K., et al., Chaogates: Morphing Logic Gates That Exploit Dynamical Patterns, *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 2010, vol. 20, no. 3, article no. 037107.
19. Smith, A., Simple Computation-Universal Cellular Spaces, *Journal of the ACM*, 1971, vol. 18, no. 3, pp. 339–353.
20. Siegelmann, H. and Sontag, E., Turing Computability with Neural Nets, *Appl. Math. Lett.*, 1991, vol. 4, no. 6, pp. 77–80.
21. Munakata, T., Sinha, S., and Ditto, W.L., Chaos Computing: Implementation of Fundamental Logical Gates by Chaotic Elements, *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 2002, vol. 49, no. 11, pp. 1629–1633.
22. Adamatzky, A., *Collision-Based Computing*, London: Springer-Verlag, 2002.
23. Durand-Lose, J., About the Universality of the Billiard Ball Model, *Multiple-Valued Logic*, 1998, vol. 6, no. 5, pp. 118–132.
24. Adamatzky, A., Programming Reaction-Diffusion Processors, in *Unconventional Programming Paradigms*, Berlin–Heidelberg: Springer, 2005, pp. 33–46.
25. Gorecki, J., Gizynski, K., Guzowski, J., et al., Chemical Computing with Reaction-Diffusion Processes, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 2015, vol. 373, no. 2046, pp. 20140219.
26. Adamatzky, A., de Lacy Costello, B., and Shirakawa, T., Universal Computation with Limited Resources: Belousov–Zhabotinsky and Physarum Computers, *International Journal of Bifurcation and Chaos*, 2008, vol. 18, no. 8, pp. 2373–2389.
27. Alhazov, A., Margenstern, M., Rogozhin, V., et al., Communicative P Systems with Minimal Cooperation, *The International Workshop on Membrane Computing* (*WMC5*), Milan, 2004. Revised papers published in *LNCS*, vol. 3365, Berlin: Springer, 2005, pp. 162–178.
28. Freund, R., Kari, L., Oswald, M., and Sosik, P., Computationally Universal P Systems without Priorities: Two Catalysts Are Sufficient, *Theoretical Computer Science*, 2005, vol. 330, no. 2, pp. 251–266.
29. Aman, B. and Ciobanu, G., Turing Completeness Using Three Mobile Membranes, in *Unconventional Computation* (*UC 2009*), Calude, C.S., Costa, J.F., Dershowitz, N., Freire, E., and Rozenberg, G., Eds., *Lecture Notes in Computer Science*, vol. 5715, Berlin–Heidelberg: Springer-Verlag, 2009, pp. 42–55.
30. Lindenmayer, A., Mathematical Models for Cellular Interaction in Development I-II, *J. Theoret. Biology*, 1968, no. 18, pp. 280–315.
31. Goel, N. and Rozehnal, I., A High-Level Language for L-systems and Its Applications, in *Lindenmayer Systems: Impacts on Theoretical Computer Science, Computer Graphics, and Developmental Biology*, Rozenberg, G. and Salomaa, A., Eds., Berlin: Springer-Verlag, 1992, pp. 231–251.

32. Winfree, E., Yang, X., and Seeman, N.C. Universal Computation via Self-assembly of DNA: Some Theory and Experiments, *Proceedings of the 2nd DIMACS Workshop on DNA Based Computers*, June 10–12, 1996, Princeton, 1996, pp. 191–213.
33. Currin, A., Korovin, K., Ababi, M., et al., Computing Exponentially Faster: Implementing a Non-Deterministic Universal Turing Machine Using DNA, *Journal of the Royal Society. Interface*, 2017, vol. 14, no. 128, article no. 20160990.
34. Zhu, L., Kim, S.-J., Hara, M., and Aono, M., Remarkable Problem-Solving Ability of Unicellular Amoeboid Organism and Its Mechanism, *Royal Society Open Science*, 2018, vol. 5, no. 12, article no. 180396.
35. Tarakanov, A. and Dasgupta, D., A Formal Model of an Artificial Immune System, *Biosystems*, 2000, vol. 55, no. 1-3, pp. 151–158.
36. Gong, M., Jiao, L., and Zhang, X., A Population-Based Artificial Immune System for Numerical Optimization, *Neurocomputing*, 2008, vol. 72, no. 1–3, pp. 149–161.
37. Ülker, E. and Arslan, A., Automatic Knot Adjustment Using an Artificial Immune System for B-Spline Curve Approximation, *Information Sciences*, 2009, vol. 179, no. 10, pp. 1483–1494.
38. Chakraborty, A. and Kar, A.K., Swarm Intelligence: A Review of Algorithms, in *Nature-Inspired Computing and Optimization. Modeling and Optimization in Science and Technologies*, Patnaik, S., Yang, X.S., and Nakamatsu, K., Eds., vol. 10, Cham: Springer, 2017, pp. 475–494.
39. Wiedermann, J., Computability and Non-computability Issues in Amorphous Computing, in *Theoretical Computer Science* (*TCS 2012*), *Lecture Notes in Computer Science*, Baeten, J.C.M., Ball, T., and de Boer, F.S., Eds., vol. 7604, Berlin– Heidelberg: Springer, 2012, pp. 1–9.
40. Wiedermann, J. and Petrů, L., On the Universal Computing Power of Amorphous Computing Systems, *Theory of Computing Systems*, 2009, vol. 45, no. 4, pp. 995–1010.
41. Hedlund, G.A., Endomorphisms and Automorphisms of the Shift Dynamical System, *Mathematical Systems Theory*, 1969, vol. 3, no. 4, pp. 320–375.
42. Lee, Y.C., Qian, S., Jones, R.D., et al., Adaptive Stochastic Cellular Automata: Theory, *Physica D: Nonlinear Phenomena*, 1990, vol. 45, no. 1–3, pp. 159–180.
43. Kleene, S.C., Representation of Events in Nerve Nets and Finite Automata, in *Automata Studies* (AM-34), vol. 34, Princeton: Princeton University Press, 1956, pp. 3–42.
44. Rosenblatt, F., The Perceptron – a Perceiving and Recognizing Automaton, *Report 85–460–1*, Cornell Aeronautical Laboratory, 1957.
45. Schuman, C.D, Potok, T.E., Patton, R.M., et al., A Survey of Neuromorphic Computing and Neural Networks in Hardware, *arXiv:1705.06963*, 2017.
46. Ditto, W.L., Sinha, S., and Murali, K., Method and Apparatus for a Chaotic Computing Module Using Threshold Reference Signal Implementations, US Patent WO/2005/036353, 2006.
47. Adamatzky, A. and Durand-Lose, J., Collision-Based Computing, in *Handbook of Natural Computing*, Rozenberg, G., et al., Eds., Berlin–Heidelberg: Springer-Verlag, 2012, pp. 1950–1978.
48. Toth, R., Stone, C., de Lacy Costello, B., et al., Simple Collision-Based Chemical Logic Gates with Adaptive Computing, *International Journal of Nanotechnology and Molecular Computation*, 2009, vol. 1, no. 3, pp. 1–16.
49. Kolmogorov A.N., Petrovskii I.G., and Piskunov, N.S., Examining the Equation of a Diffusion Related to an Increase in Matter and Its Application to a Biological Problem, *Byull. Mosk. Gos. Univ. Ser. A. Mat. Mekh.*, 1937, no. 1, pp. 1–26. (In Russian.)
50. Vanag, V.K. and Epstein, I.R., Stationary and Oscillatory Localized Patterns, and Subcritical Bifurcations, *Physical Review Letters*, 2004, vol. 92, no. 12, article no. 128301.
51. Hoffmann, M, Fröhner, C, and Noé, F., ReaDDy 2: Fast and Flexible Software Framework for Interacting-Particle Reaction Dynamics, *PLOS Computational Biology*, 2019, vol. 15, no. 2, article no. e1006830.
52. Garcia-Quismondo, M., Gutiérrez-Escudero, R., Martínez-del-Amor, M.A., et al., P-Lingua 2.0: A Software Framework for Cell-Like P Systems, *International Journal of Computers, Communications & Control*, 2009, vol. 4, no. 3, pp. 234–243.
53. Dasgupta, D., Artificial Neural Networks and Artificial Immune Systems: Similarities and Differences, *Proceedings of IEEE 1997 International Conference on Computational Cybernetics and Simulation: Systems, Man, and Cybernetics*, vol. 1, Orlando, 1997, pp. 873–878.
54. Birge, B., PSOt – a Particle Swarm Optimization Toolbox for Use with Matlab, *Proceedings of the 2003 IEEE Swarm Intelligence Symposium* (*SIS'03*), Indianapolis, 2003, pp. 182–186.
55. Qi, R., Hu, B., and Cournède, P., PSOTS: A Particle Swarm Optimization Toolbox in Scilab, *Proceedings of IEEE 2009 International Workshop on Open-source Software for Scientific Computation* (*OSSC*), Guiyang, 2009, pp. 107–114.
56. Woods, D., Doty, D., Myhrvold, C., et al., Diverse and Robust Molecular Algorithms Using Repro-Grammable DNA Self-Assembly, *Nature*, 2019, vol. 567, no. 7748, pp. 366–372.
57. Hunt, J., Timmis, J., Cooke, D., et al., Jisys: Development of an Artificial Immune System for Real World Applications, in *Artificial Immune Systems and Their Applications*, Dasgupta, D., Ed., Berlin: Springer-Verlag, 1998, pp. 157–186.
58. Shen, X, Gao, X.Z., Bie, R, and Jin, X., Artificial Immune Networks: Models and Applications, *Proceedings of the International Conference on Computational Intelligence and Security* (*ICCIAS 2006*), Guangzhou, 2006, pp. 394–397.
59. Twycross, J. amd Aickelin, U., Libtissue - Implementing Innate Immunity, *Proceedings of IEEE 2006 International Conference on Evolutionary Computation* (*CEC*), Vancouver, 2006, pp. 499–506.
60. Coore, D., Botanical Computing: A Developmental Approach to Generating Interconnect Topologies on an Amorphous Computing, *PhD thesis*, MIT, 1999.
61. Beal, J. and Bachrach, J., Infrastructure for Engineered Emergence on Sensor/Actuator Networks, *IEEE Intelligent Systems*, 2006, vol. 21, no. 2, pp. 10–19.
62. Secco, J., Farina, M., Demarchi, D., et al., Memristor Cellular Automata for Image Pattern Recognition and Clinical Applications, *Proceedings of IEEE 2016 International Symposium on Circuits and Systems* (*ISCAS*), Montreal, 2016, pp. 1378–1381.
63. Miranda, G., Machicao, J., and Bruno, O., Exploring Spatiotemporal Dynamics of Cellular Automata for Pattern Recognition in Networks, *Scientific Reports*, 2016, vol. 6, pp. 37329–37329-15.
64. Partanen, J., An Urban Cellular Automata Model for Simulating Dynamic States on a Local Scale, *Entropy*, 2017, vol. 19, no. 1, article no. 12.
65. Das, A.K. and Chattaraj, U., Heterogeneous Traffic Simulation for Urban Streets Using Cellular Automata, *Arabian Journal for Science and Engineering*, 2019, vol. 44, no. 10, pp. 8557–8571.
66. Sree, P.K. and Babu, I.R., Towards a Cellular Automata Based Network Intrusion Detection System with Power Level Metric in Wireless Adhoc Networks (IDFADNWCA), *Proceedings of*

the 2008 International Conference on Advanced Computer Theory and Engineering (*ICACTE*), Phuket, 2008, pp. 1071–1075.
67. Nisha, V.M. and Jeganathan, L., A Symmetry Based Anomaly Detection in Brain Using Cellular Automata for Computer Aided Diagnosis, *Indonesian Journal of Electrical Engineering and Computer Science*, 2019, vol. 14, no. 1, pp. 471–477.
68. Kabli, F., Hamou, R.M., and Amine, A., DNA Data Clustering by Combination of 3D Cellular Automata and N-Grams for Structure Molecule Prediction, *International Journal of Bioinformatics Research and Applications*, 2016, vol. 12, no. 4, pp. 299–311.
69. Berberoğlu, S., Akın, A., and Clarke, K.C., Cellular Automata Modeling Approaches to Forecast Urban Growth for Adana, Turkey: A Comparative Approach, *Landscape and Urban Planning*, 2016, vol. 153, pp. 11–27.
70. Shen, L., Shen, Y., Song, C., et al., A Novel Power System Anomaly Data Identification Method Based on Neural Network and Affine Propagation, *Proceedings of the International Conference on Artificial Intelligence and Security* (*ICAIS*), New York, 2019, pp. 499–508.
71. Ishitaki, T., Oda, T., and Barolli, L., A Neural Network Based User Identification for Tor Networks: Data Analysis Using Friedman Test, *Proceedings of the IEEE 2016 30th International Conference on Advanced Information Networking and Applications Workshops* (*WAINA*), Crans-Montana, 2016, pp. 7–13.
72. Tobiyama, S., Yamaguchi, Y., Shimada, H., et al., Malware Detection with Deep Neural Network Using Process Behavior, *Proceedings of the IEEE 40th Annual Computer Software and Applications Conference* (*COMPSAC*), vol. 2, Atlanta, 2016, pp. 577–582.
73. Manoj, K. and Charul, B., Hybrid Tracking Model and GSLM Based Neural Network for Crowd Behavior Recognition, *Journal of Central South University*, 2017, vol. 24, no. 9, pp. 2071–2081.
74. Raman, M.G., Somu, N., and Mathur, A.P., Anomaly Detection in Critical Infrastructure Using Probabilistic Neural Network, *Proceedings of the International Conference on Applications and Techniques in Information Security*, Tamil Nadu, 2019, pp. 129–141.
75. Yuan, F., Cao, Y., and Shang, Y., Insider Threat Detection with Deep Neural Network, *Proceedings of the International Conference on Computational Science*, Wuxi, 2018, pp. 43–54.
76. Jiang, Y., Li, C.H, Yu, L.S., and Bao, B., On Network Security Situation Prediction Based on RBF Neural Network, *Proceedings of the IEEE 2017 36th Chinese Control Conference* (*CCC*), Liaoning, 2017, pp. 4060–4063.
77. Pang, Y., Xue, X., and Wang, H., Predicting Vulnerable Software Components through Deep Neural Network, *Proceedings of the 2017 International Conference on Deep Learning Technologies*, Chengdu, 2017, pp. 6–10.
78. Poteralski, A., Hybrid Artificial Immune Strategy in Identification and Optimization of Mechanical Systems, *Journal of Computational Science*, 2017, vol. 23, pp. 216–225.
79. Lima, F.P, Chavarette, F.R, Souza, S.S, and Lopes, M.L., Monitoring and Fault Identification in Aeronautical Structures Using an Wavelet-Artificial Immune System Algorithm, in *Probabilistic Prognostics and Health Management of Energy Systems*, Ekwaro-Osire, S., Gonçalves, A., and Alemayehu, F., Eds., Cham: Springer, 2017, pp. 203–219.
80. Purbasari, A., Supriana, I., Santoso, O.S., and Mandala, R., Designing Artificial Immune System Based on Clonal Selection: Using Agent-Based Modeling Approach, *Proceedings of the IEEE 2013 7th Asia Modelling Symposium*, Hong Kong, 2013, pp. 11–15.
81. Liu, Y., Ding, Y., Hao, K., and Chen, L., An Immune System-Inspired Information Diffusion Model, *Proceedings of the IEEE 2017 36th Chinese Control Conference* (*CCC*), Liaoning, 2017, pp. 11238–11243.
82. Vasilyev, V. and Shamsutdinov, R., Distributed Intelligent System of Network Traffic Anomaly Detection Based on Artificial Immune System, *Proceedings of the 7th Scientific Conference on Information Technologies for Intelligent Decision Making Support* (*ITIDS 2019*), Ufa: Atlantis Press, 2019, pp. 40–45.
83. Jiang, Q. and Chang, F., A Novel Antibody Population Optimization Based Artificial Immune System for Rotating Equipment Anomaly Detection, *Journal of Mechanical Science and Technology*, 2020, vol. 34, no. 9, pp. 3565–3574.
84. Wang, M., Ge, J., Zhang, D., and Zhang, F., An Improved Artificial Immune System Model for Link Prediction, *Proceedings of the Pacific Rim International Conference on Artificial Intelligence* (*PRICAI*), Nanjing, 2018, pp. 1–9.
85. Günay, M., Orman, Z., Ensari, T., et al., Diagnosis of Lung Cancer Using Artificial Immune System, *Proceedings of IEEE 2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science* (*EBBT*), Istanbul, 2019, pp. 1–4.
86. Maji, P., Shaw, C., Ganguly, N., et al., Theory and Application of Cellular Automata for Pattern Classification, *Fundamenta Informaticae*, 2003, vol. 58, no. 3–4, pp. 321–354.
87. Zhou, L. and Yang, M., A Classifier Build around Cellular Automata for Distributed Data Mining, *Proceedings of the IEEE 2008 International Conference on Computer Science and Software Engineering*, vol. 4, Wuhan, 2008, pp. 312–315.
88. Chang, H., Baek, S., Kim, H., et al., Development of Distributed Real-Time Decision Support System for Traffic Management Centers Using Microscopic CA Model, *Iranian Journal of Science & Technology, Transaction B, Engineering*, 2007, vol. 31, no. B2, pp. 155–166.
89. Benhacine, F.Z., Atmani, B., Benamina, M., et al., A Visual Decision Making Support System for the Diabetes Prevention, in *Advances in Data Science, Cyber Security and IT Applications*, Alfaries, A., Mengash, H., Yasar, A., and Shakshuki, E., Eds., *Proceedings of the 1st International Conference on Computing* (*ICC 2019*), Part II, Riyadh, 2019, pp. 81–92.
90. Liu, Y. and He, J., Developing a Web-Based Cellular Automata Model for Urban Growth Simulation, *Proceedings of the International Symposium on Spatial Analysis, Spatial-Temporal Data Modeling, and Data Mining*, SPIE, vol. 7492, Wuhan, 2009, article no. 74925C.
91. Wainer, G., Developing a Software Toolkit for Urban Traffic Modeling, *Software: Practice and Experience*, 2007, vol. 37, no. 13, pp. 1377–1404.
92. Li, Y., Zhang, H., and Shen, Q., Spectral–Spatial Classification of Hyperspectral Imagery with 3D Convolutional Neural Network, *Remote Sensing*, 2017, vol. 9, no. 1, pp. 67– 88.
93. Jia, F., Lei, Y., Lu, N., and Xing, S., Deep Normalized Convolutional Neural Network for Imbalanced Fault Classification of Machinery and Its Understanding via Visualization, *Mechanical Systems and Signal Processing*, 2018, vol. 110, pp. 249–367.
94. Beşikçi, E.B., Arslan, O., Turan, O., and Ölçer, A.I., An Artificial Neural Network Based Decision Support System for Energy Efficient Ship Operations, *Computers & Operations Research*, 2016, vol. 66, pp. 393–401.
95. Boonpeng, S. and Jeatrakul, P., Decision Support System for Investing in Stock Market by Using OAA-Neural Network, *Proceedings of the IEEE 2016 8th International Conference on Advanced Computational Intelligence* (*ICACI*), Chiang Mai, 2016, pp. 1–6.

96. Tang, G. and Zeng, H., Chemical Production Information Management System Based on Artificial Intelligence Neural Network Algorithm, *Chemical Engineering Transactions*, 2018, vol. 66, pp. 967–972.
97. Elechi, P., Improved Ghost Worker Fraud Detection System Using Artificial Neural Network, *Journal of Electrical Engineering, Electronics, Control and Computer Science*, 2019, vol. 5, no. 1, pp. 17–24.
98. Magna, G., Casti, P., Jayaraman, S.V., et al., Identification of Mammography Anomalies for Breast Cancer Detection by an Ensemble of Classification Models Based on Artificial Immune System, *Knowledge-Based Systems*, 2016, vol. 101, pp. 60–70.
99. Aldhaheri, S., Alghazzawi, D., Cheng, L., et al., DeepDCA: Novel Network-Based Detection of IoT Attacks Using Artificial Immune System, *Applied Sciences*, 2020, vol. 10, no. 6, pp. 1909–1909-23.
100. Mnif, S., Elkosantini, S., Darmoul, S., and Said, L.B., An Immune Multi-agent Based Decision Support System for the Control of Public Transportation Systems, *Proceedings of the International Conference on Practical Applications of Agents and Multi-Agent Systems*, Seville, 2016, pp. 187–198.
101. Berquedich, M., Kamach, O., Masmoudi, M., and Deshayes, L., Agile Decision Support System for the Management of Tensions in Emergency Services Using AIS Techniques, *Proceedings of the 2017 IEEE International Colloquium on Logistics and Supply Chain Management* (*LOGISTIQUA*), Rabat, 2017, pp. 118–123.
102. do Nascimento Alves, H., Machado, R.C., and Bergê, I.G., Design and Development of a Software for Fault Diagnosis in Radial Distribution Networks, *Proceedings of the IEEE/IAS 9th International Conference on Industry Applications* (*INDUSCON 2010*), São Paulo, 2010, pp. 1–6.
103. Chitra, M.E. and Rajaram, M., A Software Reliability Estimation Tool Using Artificial Immune Recognition System, *Proceedings of the International MultiConference of Engineers and Computer Scientists* (*IMECS 2008*), vol. 1, Hong Kong, 2008, pp. 967–975.

This paper was recommended for publication by V.V. Kul'ba, a member of the Editorial Board.

Received January 28, 2021, and revised March 31, 2021.
Accepted April 6, 2021.

Author information

Shiroky, Aleksandr Aleksandrovich. Cand. Sci. (Phys.–Math.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia, ✉ shiroky@ipu.ru

Kalashnikov, Andrei Olegovich. Dr. Sci. (Eng.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia, ✉ aokalash@ipu.ru

Cite this article

Shiroky, A.A., Kalashnikov, A.O., Natural Computing with Application to Risk Management in Complex Systems. *Control Sciences* **4,** 2–17 (2021). http://doi.org/10.25728/cs.2021.4.1

Original Russian Text © Shiroky, A.A., Kalashnikov, A.O., 2021, published in *Problemy Upravleniya*, 2021, no. 4, pp. 3–20.

DOI: http://doi.org/10.25728/cs.2021.4.2

A PARAMETRIC LYAPUNOV FUNCTION FOR DISCRETE-TIME CONTROL SYSTEMS WITH BOUNDED EXOGENOUS DISTURBANCES: ANALYSIS

M.V. Khlebnikov[1] and Ya.I. Kvinto[2]

Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia

[1]✉ khlebnik@ipu.ru, [2]✉ yanakvinto@mail.ru

Abstract. This paper considers a linear discrete-time dynamic system subjected to arbitrary bounded exogenous disturbances described by a matrix from a convex affine family. A simple approach to designing a parametric quadratic Lyapunov function for this system is proposed. It involves linear matrix inequalities and a fruitful technique to separate the system matrix and the Lyapunov function matrix in the matrix inequality expressing a stability condition of the system. Being well known, this technique, however, has not been previously applied to dynamic systems with nonrandom bounded exogenous disturbances. According to the numerical simulations, the parametric quadratic Lyapunov function-based approach yields appreciably less conservative results for the class of systems under consideration than the common quadratic Lyapunov function-based one.

Keywords: dynamic system, linear discrete-time system, parametric quadratic Lyapunov function, common quadratic Lyapunov function, bounded exogenous disturbances, robustness, linear matrix inequalities, analysis problem, conservatism, structured matrix uncertainty.

INTRODUCTION

The study of dynamic systems with parametric uncertainty and exogenous disturbances is of great theoretical and practical interest. A classical approach to solving this class of problems is based on designing a common quadratic Lyapunov function for the entire family of systems [1–4], and the apparatus of linear matrix inequalities [5] is a convenient technique.

However, as is well known [6], using a common Lyapunov function often yields rather conservative results. In this regard, let us consider the problem of designing a parametric Lyapunov function for continuous- and discrete-time systems with uncertainty. In the papers [6–8], the advantages of a parametric quadratic Lyapunov function were presented; as shown therein, this approach decreases the solution's degree of conservatism compared to the common quadratic Lyapunov function-based one. Among the relatively recent publications on the subject, we mention, e.g., [1, 9–11]. In the paper [8], an efficient method was proposed for designing a parametric quadratic Lyapunov function using linear matrix inequalities when analyzing the stability of an affine family of continuous-time systems. Later on [7], this approach was extended to the case of discrete-time systems with parametric uncertainty. In the paper [12], the result of [7] was generalized to the case of a discrete-time system with parametric and structured matrix uncertainty.

In this paper, we design a parametric quadratic Lyapunov function for an affine family of discrete-time systems subjected to arbitrary bounded exogenous disturbances. Such problems often arise in applications and have a transparent physical motivation; for example, see [9, 13]. As an important technique, we employ and generalize an equivalent representation of the system's stability condition in the form of a matrix inequality, first proposed in [7] (also, see [14]). This representation allows separating the system matrix and the Lyapunov function matrix in the matrix inequality. The technique gave rise to several further generalizations, but it has not been previously applied to dynamic systems with nonrandom bounded exogenous disturbances. The corresponding assertion (see Theorem 1 below) is

novel, like Theorem 2 obtained on its basis. According to the numerical simulations, the parametric quadratic Lyapunov function-based approach yields appreciably less conservative results for the class of systems under consideration than the common quadratic Lyapunov function-based one.

A rather close problem was examined in [6], where new sufficient conditions for robust quadratic stability of a discrete-time system with parametric uncertainty were established. The problem statement introduced below is more general, and the apparatus of linear matrix inequalities is used as a convenient technique.

This paper is organized as follows. Section 1 describes the problem statement and the approach to solve it. The main results are formulated in Section 2. The numerical simulation results are presented and discussed in Section 3.

Throughout the paper, $\|\cdot\|$ denotes the Euclidean norm of vectors and the spectral norm of matrices, the symbol $^{\mathrm{T}}$ indicates transposition, I is an identity matrix of compatible dimensions, and all matrix inequalities are understood in the sense of positive or negative (semi-) definiteness of the corresponding matrices.

1. PROBLEM STATEMENT AND SOLUTION APPROACHES

Consider a linear discrete-time dynamic system described by

$$x_{k+1} = A(\alpha)x_k + Dw_k, \qquad (1)$$

where $x_k \in \mathbb{R}^n$ denotes the state vector, x_0 is an initial condition, and $w_k \in \mathbb{R}^m$ is an exogenous disturbance satisfying the constraint

$$\| w_k \| \le 1, \quad k = 1, 2, \ldots \qquad (2)$$

Let $D \in \mathbb{R}^{n\times m}$ and the matrices $A(\alpha) \in \mathbb{R}^{n\times n}$ belong to the convex family

$$\mathbb{A} = \left\{ A(\alpha):\ A(\alpha) = \sum_{i=1}^{N} \alpha_i A_i,\ \sum_{i=1}^{N} \alpha_i = 1,\ \alpha_i \ge 0 \right\}. \qquad (3)$$

Assume that the system (1) is stable: all matrices $A(\alpha) \in \mathbb{A}$ are Schur (their eigenvalues lie inside the unit circle), and the pair (A, D) is controllable.

The main problem is to design a parametric quadratic Lyapunov function for the system (1) and (2).

First of all, we discuss an approach to designing a parametric quadratic Lyapunov function for the dynamic system

$$x_{k+1} = Ax_k + Dw_k, \ \| w_k \| \le 1, \ k = 1, 2, \ldots, \qquad (4)$$

with matrices $A \in \mathbb{R}^{n\times n}$ and $D \in \mathbb{R}^{n\times m}$, the state vector $x_k \in \mathbb{R}^n$, an initial condition x_0, and an exogenous disturbance $w_k \in \mathbb{R}^m$ (2).

According to the paper [15] (also, see the monograph [16]), a matrix $0 < P \in \mathbb{R}^{n\times n}$ satisfying the linear matrix inequality

$$\frac{1}{\mu} APA^{\mathrm{T}} - P + \frac{1}{1-\mu} DD^{\mathrm{T}} \le 0 \qquad (5)$$

for some $0 < \mu < 1$ will define a quadratic Lyapunov function of the form

$$V(x) = x^{\mathrm{T}} P^{-1} x$$

for the system (4) and (2).

For further presentation, we need the following technical result.

Theorem 1. *The assertions below are equivalent:*

I. *There exists a matrix* $P > 0$ *such that*

$$\frac{1}{\mu} APA^{\mathrm{T}} - P + \frac{1}{1-\mu} DD^{\mathrm{T}} \le 0 \qquad (6)$$

for some $0 < \mu < 1$.

II. *There exist matrices* $0 < P = P^{\mathrm{T}} \in \mathbb{R}^{n\times n}$ *and* $G \in \mathbb{R}^{n\times n}$ *such that*

$$\begin{pmatrix} P & AG & D \\ G^{\mathrm{T}}A^{\mathrm{T}} & \mu\left(G + G^{\mathrm{T}} - P\right) & 0 \\ D^{\mathrm{T}} & 0 & (1-\mu)I \end{pmatrix} \ge 0 \qquad (7)$$

for some $0 < \mu < 1$.

P r o o f. Applying Schur's complement lemma to the matrix inequality (7) two sequential times gives the equivalent relations

$$\begin{pmatrix} P - \dfrac{1}{1-\mu} DD^{\mathrm{T}} & AG \\ G^{\mathrm{T}}A^{\mathrm{T}} & \mu\left(G + G^{\mathrm{T}} - P\right) \end{pmatrix} \ge 0 \qquad (8)$$

and

$$P - \frac{1}{1-\mu} DD^{\mathrm{T}} - \frac{1}{\mu} AG(G + G^{\mathrm{T}} - P)^{-1} G^{\mathrm{T}} A^{\mathrm{T}} \ge 0. \qquad (9)$$

Letting $G = G^{\mathrm{T}} = P$ in (9), we arrive at inequality (6). Thus, Assertion I implies Assertion II.

Now, we prove the converse. Multiplying (8) on the left by $\begin{pmatrix} I & -\dfrac{1}{\mu}A \end{pmatrix}$ and on the right by $\begin{pmatrix} I \\ -\dfrac{1}{\mu}A^{\mathrm{T}} \end{pmatrix}$ yields

$$\begin{pmatrix} I & -\dfrac{1}{\mu}A \end{pmatrix} \begin{pmatrix} P - \dfrac{1}{1-\mu} DD^{\mathrm{T}} & AG \\ G^{\mathrm{T}}A^{\mathrm{T}} & \mu\left(G + G^{\mathrm{T}} - P\right) \end{pmatrix} \begin{pmatrix} I \\ -\dfrac{1}{\mu}A^{\mathrm{T}} \end{pmatrix} \ge 0$$

or $P-\frac{1}{1-\mu}DD^{\mathrm{T}}-\frac{1}{\mu}AG^{\mathrm{T}}A^{\mathrm{T}}-\frac{1}{\mu}AGA^{\mathrm{T}}+\frac{1}{\mu}A\left(G+G^{\mathrm{T}}-P\right)A^{\mathrm{T}}\geq 0$. This condition is equivalent to (6). The proof of Theorem 1 is complete. ♦

Well, Theorem 1 reduces inequality (5) to the equivalent form (7), linear in the matrix variables P and A.

Let us return to the system (1). Due to the convexity of (3), a solution $P>0$ of the combined matrix inequalities

$$\frac{1}{\mu}A_iPA_i^{\mathrm{T}}-P+\frac{1}{1-\mu}DD^{\mathrm{T}}\leq 0,\quad 0<\mu<1,\quad i=1,\ldots,N,$$

or, by Theorem 1, of the equivalent ones

$$\begin{pmatrix} P & A_iG & D \\ G^{\mathrm{T}}A_i^{\mathrm{T}} & \mu\left(G+G^{\mathrm{T}}-P\right) & 0 \\ D^{\mathrm{T}} & 0 & (1-\mu)I \end{pmatrix}\geq 0,\quad 0<\mu<1, \tag{10}$$

will define the common quadratic Lyapunov function $V(x)=x^{\mathrm{T}}P^{-1}x$ for the affine family (1)–(3).

Note a rather high degree of conservatism for this result: an appropriate matrix $P>0$ must satisfy all inequalities (10) for the same value μ.

Appreciably less conservative results are obtained using the *parametric* quadratic Lyapunov function $V(x)=x^{\mathrm{T}}P^{-1}(\alpha)x$ with the matrix

$$P(\alpha)=\sum_{i=1}^{N}\alpha_iP_i,\quad 0<P_i=P_i^{\mathrm{T}}\in\mathbb{R}^{n\times n}. \tag{11}$$

Resting on Theorem 1, we will establish a sufficient condition for the existence of a parametric quadratic Lyapunov function (11) for the affine family (1)–(3); see the next section.

2. MAIN RESULT

Due to the convexity of (3) and the structure of the parametric quadratic Lyapunov function (11), it suffices to require that the component P_i correspond to the vertex A_i in the system (1). The following assertion is the main result of the paper.

Theorem 2. *Assume that there exist matrices* $0<P_i=P_i^{\mathrm{T}}\in\mathbb{R}^{n\times n}$ *and* $G\in\mathbb{R}^{n\times n}$ *such that*

$$\begin{pmatrix} P_i & A_iG & D \\ G^{\mathrm{T}}A_i^{\mathrm{T}} & \mu\left(G+G^{\mathrm{T}}-P_i\right) & 0 \\ D^{\mathrm{T}} & 0 & (1-\mu)I \end{pmatrix}\geq 0,\quad i=1,\ldots,N, \tag{12}$$

for some $0<\mu<1$.

Then the system (1), (3), and (2) *has a parametric quadratic Lyapunov function with the matrix*

$$P(\alpha)=\sum_{i=1}^{N}\alpha_iP_i. \tag{13}$$

P r o o f. Due to (5), a matrix $P_i>0$ satisfying the matrix inequality

$$\frac{1}{\mu}A_iP_iA_i^{\mathrm{T}}-P_i+\frac{1}{1-\mu}DD^{\mathrm{T}}\leq 0 \tag{14}$$

for some $0<\mu<1$ will define the quadratic Lyapunov function $V(x)=x^{\mathrm{T}}P_i^{-1}x$ for the system (1) at the vertex A_i.

According to Theorem 2, condition (14) is equivalent to the matrix inequality

$$\begin{pmatrix} P_i & A_iG & D \\ G^{\mathrm{T}}A_i^{\mathrm{T}} & \mu\left(G+G^{\mathrm{T}}-P_i\right) & 0 \\ D^{\mathrm{T}} & 0 & (1-\mu)I \end{pmatrix}\geq 0 \tag{15}$$

for some $0<\mu<1$.

Multiplying (15) by α_i and summing over $i=1,\ldots,N$, we obtain

$$\begin{pmatrix} \sum\limits_{i=1}^{N}\alpha_iP_i & \left(\sum\limits_{i=1}^{N}\alpha_iA_i\right)G & \sum\limits_{i=1}^{N}\alpha_iD \\ G^{\mathrm{T}}\sum\limits_{i=1}^{N}\alpha_iA_i^{\mathrm{T}} & \mu\left(\sum\limits_{i=1}^{N}\alpha_i\left(G+G^{\mathrm{T}}\right)-\sum\limits_{i=1}^{N}\alpha_iP_i0\right) & \\ \sum\limits_{i=1}^{N}\alpha_iD^{\mathrm{T}} & 0 & \sum\limits_{i=1}^{N}\alpha_i(1-\mu)I \end{pmatrix}\geq 0.$$

Since $\sum\limits_{i=1}^{N}\alpha_iA_i=A(\alpha)$ and $\sum\limits_{i=1}^{N}\alpha_i=1$,

$$\begin{pmatrix} \sum\limits_{i=1}^{N}\alpha_iP_i & A(\alpha)G & D \\ G^{\mathrm{T}}A^{\mathrm{T}}(\alpha) & \mu\left(G+G^{\mathrm{T}}-\sum\limits_{i=1}^{N}\alpha_iP_i0\right) & \\ D^{\mathrm{T}} & 0 & (1-\mu)I \end{pmatrix}\geq 0.$$

Thus, the matrix (13) defines the parametric quadratic Lyapunov function $V(x)=x^{\mathrm{T}}P^{-1}(\alpha)x$ for the system (1)–(3). The proof of Theorem 2 is complete. ♦

Clearly, this approach is conservative primarily because conditions (12) must hold for the same value μ. However, as we will see below, the proposed approach yields less conservative estimates than the common quadratic Lyapunov function-based one.

3. AN EXAMPLE

As an illustrative example, we consider the system from [4] with the vertices

$$A_1=\begin{pmatrix} 0.0061 & -0.2630 & 0.2748 \\ 0.1266 & 0.1242 & -0.3029 \\ -0.5100 & 0.4678 & -0.9712 \end{pmatrix},$$

$$A_2 = \begin{pmatrix} 0.1330 & 0.2009 & 0.1672 \\ 0.1224 & -0.5987 & 0.3100 \\ -0.5235 & 0.0297 & -0.4784 \end{pmatrix},$$

$$A_3 = \begin{pmatrix} -0.2733 & -0.1868 & -0.0077 \\ -0.0253 & -0.2828 & 0.6112 \\ -0.2412 & -0.0844 & -0.8024 \end{pmatrix},$$

and the bounded exogenous disturbances with

$$D = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}.$$

Using Theorem 2, we solve the corresponding optimization problem $\min \sum_i \| P_i \|$ and find the matrices

$$\hat{P}_1 = \begin{pmatrix} 1.1381 & 0.8630 & -0.2336 \\ 0.8630 & 1.1608 & 0.2290 \\ -0.2336 & 0.2290 & 0.8764 \end{pmatrix} \cdot 10^4,$$

$$\hat{P}_2 = \begin{pmatrix} 1.1552 & 0.8149 & -0.2110 \\ 0.8149 & 1.3914 & 0.1546 \\ -0.2110 & 0.1546 & 0.3824 \end{pmatrix} \cdot 10^4,$$

and $\hat{P}_3 = \begin{pmatrix} 1.2724 & 1.1669 & 0.1901 \\ 1.1669 & 1.4113 & -0.2095 \\ 0.1901 & -0.2095 & 0.5410 \end{pmatrix} \cdot 10^4$

of the parametric quadratic Lyapunov function $V(x) = x^{\mathrm{T}} \left(\sum_{i=1}^{3} \alpha_i \hat{P}_i \right) x$.

For comparison, we calculate the matrix of the common quadratic Lyapunov function for this system by solving the optimization problem $\min \| P \|$ subject to the constraint (10):

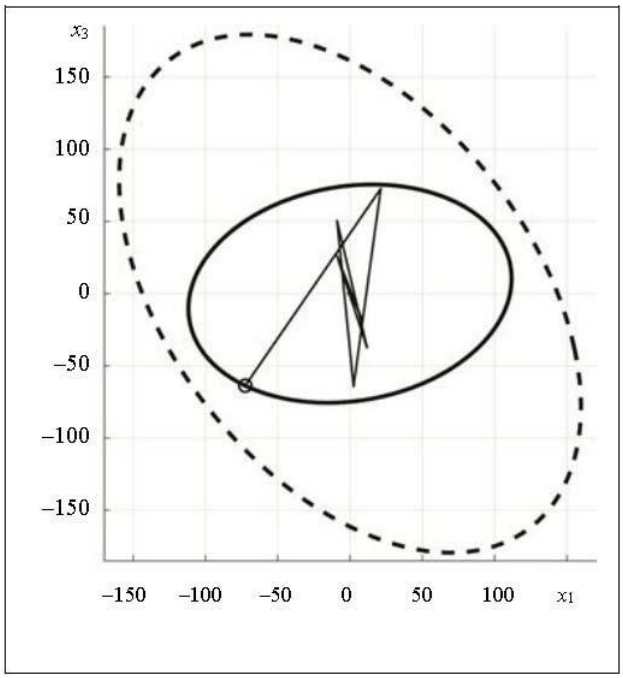
$$\hat{P} = \begin{pmatrix} 2.5519 & -0.3322 & -1.2518 \\ -0.3322 & 4.7636 & 0.5922 \\ -1.2518 & 0.5922 & 3.2176 \end{pmatrix} \cdot 10^4.$$

As is well known, the matrix of a quadratic Lyapunov function is associated with the so-called invariant ellipsoid. (For more details, see the monograph [16].) Recall that the system's trajectory starting at a point inside the invariant ellipsoid will remain there under all admissible exogenous disturbances.

Let us compare the invariant ellipsoids determined by the parametric and common quadratic Lyapunov functions. The figure shows the projections of the corresponding invariant ellipsoids onto the plane (x_1, x_3). The noticeable differences confirm the reduced degree of conservatism of the proposed approach.

Also, this figure shows the projection of the system's trajectory with an initial condition from the invariant ellipsoid with the matrix $\hat{P}(\alpha)$ under the so-called worst exogenous disturbance [16] of the form

$$\tilde{w}_k = \operatorname{sign}\left(D^{\mathrm{T}} \hat{P}^{-1}(\alpha) A(\alpha) x_k \right), \quad k = 1, 2, \ldots$$



Projections of invariant ellipsoids and system's trajectories

The calculations were performed in Matlab using cvx [17].

CONCLUSIONS

This paper has proposed an approach to designing a parametric quadratic Lyapunov function for an affine family of discrete-time systems with arbitrary bounded exogenous disturbances. The approach is remarkable for simplicity and a reduced degree of conservatism, as the numerical simulations show.

Further research will focus on extending these results to the design problem for a family of discrete-time control systems with parametric uncertainty.

REFERENCES

1. Ebihara, Y., Peaucelle, D., and Arzelier, D., *S-Variable Approach to LMI-Based Robust Control*, London: Springer-Verlag, 2015.
2. Mao, W.-J. and Chu, J., Correction to "Quadratic Stability and Stabilization of Dynamic Interval Systems," *IEEE Trans. on Automatic Control*, 2006, vol. 51, no. 8, pp. 1404–1405.
3. Mao, W.-J. and Chu, J., Quadratic Stability and Stabilization of Dynamic Interval Systems, *IEEE Trans. on Automatic Control*, 2003, vol. 48, no. 6, pp. 1007–1012.
4. Pessim, P.S.P., Lacerda, M.J., and Agulhari, C.M., Parameter-Dependent Lyapunov Functions for Robust Performance of Uncertain Systems, *IFAC PapersOnLine*, 2018, vol. 51, no. 25, pp. 293–298.
5. Boyd, S., El Ghaoui, L., Feron, E., and Balakrishnan, V., *Linear Matrix Inequalities in System and Control Theory*, Philadelphia: SIAM, 1994.

6. Ramos, D.C.W. and Peres, P.L.D., A Less Conservative LMI Condition for the Robust Stability of Discrete-Time Uncertain Systems, *Systems & Control Letters*, 2001, vol. 43, pp. 371–378.
7. De Oliveira, M.C., Bernussou, J., and Geromel, J.C., A New Discrete-Time Robust Stability Condition, *Systems & Control Letters*, 1999, vol. 37, pp. 261–265.
8. Geromel, J.C., De Oliveira, M.C., and Hsu, L., LMI Characterization of Structural and Robust Stability, *Linear Algebra and Its Applications*, 1998, vol. 285, pp. 69–80.
9. Cox, P.B., Weiland, S., and Toth, R., Affine Parameter-Dependent Lyapunov Functions for LPV Systems with Affine Dependence, *arXiv:1803.11543v2*. Last revised April 5, 2018.
10. Liu, Z., Theilliol, D., Gu, F., et al., State Feedback Controller Design for Affine Parameter-Dependent LPV Systems, *IFAC PapersOnLine*, 2017, vol. 50, no. 1, pp. 9760–9765.
11. Oliveira, R.C.L.F. and Peres, P.L.D., LMI Conditions for Robust Stability Analysis Based on Polinomially Parameter-Dependent Lyapunov Functions, *Systems & Control Letters*, 2006, vol. 55, pp. 52–61.
12. Khlebnikov, M.V. and Kvinto, Y.I. Robust Stability Conditions for a Family of Linear Discrete-Time Systems Subjected to Uncertainties, *Control Sciences*, 2020, no. 5, pp. 17–21.
13. Gahinet, P., Apkarian, P., and Chilali, M., Affine Parameter-Dependent Lyapunov Functions and Real Parametric Uncertainty, *IEEE Trans. on Automatic Control*, 1996, vol. 41, no. 3, pp. 436–442.
14. Daafouz, J. and Bernussou, J., Parameter Dependent Lyapunov Functions for Discrete Time Systems with Time Varying Parametric Uncertainties, *Systems & Control Letters*, 2001, vol. 43, iss. 5, pp. 355–359.
15. Khlebnikov, M.V., Polyak, B.T., and Kuntsevich, V.M., Optimization of Linear Systems Subject to Bounded Exogenous Disturbances: The Invariant Ellipsoid Technique, *Automation and Remote Control*, 2011, vol. 72, no. 11, pp. 2227–2275.
16. Polyak, B.T., Khlebnikov, M.V., and Shcherbakov, P.S., *Upravlenie lineinymi sistemami pri vneshnikh vozmushcheniyakh: Tekhnika lineinykh matrichnykh neravenstv* (Control of Linear Systems Subjected to Exogenous Distrubances: An LMI Approach), Moscow: LENAND, 2014. (In Russian.)
17. Grant, M. and Boyd, S., CVX: Matlab Software for Disciplined Convex Programming, Version 2.1. URL: http://cvxr.com/cvx/

This paper was recommended for publication by S.A. Krasnova, a member of the Editorial Board.

Received April 30, 2021, and revised July 8, 2021.
Accepted July 14, 2021.

Author information

Khlebnikov, Mikhail Vladimirovich. Dr. Sci. (Phys.-Math.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia, ✉ khlebnik@ipu.ru

Kvinto, Yana Igorevna. Cand. Sci. (Eng.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia, ✉ yanakvinto@mail.ru

Cite this article

Khlebnikov, M.V. and Kvinto, Ya.I., A Parametric Lyapunov Function For Discrete-time Control Systems With Bounded Exogenous Disturbances: Analysis. *Control Sciences* **4,** 18–22 (2021). http://doi.org/10.25728/cs.2021.4.2

Original Russian Text © Khlebnikov, M.V., Kvinto, Ya.I., 2021, published in *Problemy Upravleniya*, 2021, no. 4, pp. 21–26.

DOI: http://doi.org/10.25728/cs.2021.4.3

DESIGNING A DOUBLE-LOOP OBSERVER TO CONTROL A SINGLE-LINK MANIPULATOR UNDER UNCERTAINTY[1]

D.V. Krasnov[2] and A.S. Antipov[3]

Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia

[2]✉ dim93kr@mail.ru, [3]✉ scholess18@mail.ru

Abstract. A single-link manipulator with an electrical actuator is considered, and a dynamic feedback control is designed for tracking a given reference signal of its angular position. The problem statement includes the following assumptions: the output (controlled) variable is not measured; the sensors are located only on the electrical drive; the mechanical subsystem has exogenous and parametric disturbances. Under the smooth disturbances, a discontinuous control law is formed in terms of the canonical input–output system written in the tracking error. For implementing this law, a double-loop observer with piecewise linear corrections is developed. In the first loop, the controlled variable is restored using an observer of the electrical subsystem. This variable, together with the reference signal, serves to design corrections in the second loop. The second observer is a replica of the virtual input–output system. It restores mixed variables––functions of the state variables, the exogenous actions, and their derivatives—to form the feedback law. The order of the observers in each loop is reduced by discarding the dynamics of the estimated variables, treated as bounded perturbations in the observation problem. A tuning procedure is proposed that allows estimating the unmeasured endogenous and exogenous signals with a required accuracy in a given time under an additive parasitic signal in the corrections. The simulation results are presented.

Keywords: electromechanical system, tracking, invariance, sliding mode, observer of states and disturbances, piecewise linear functions.

INTRODUCTION

Robotic manipulators with electrical actuators perform various tasks in mechanical engineering and other industries. Progress in modern robotics is primarily associated with the improvement of information and control systems and the use of advanced information technology and control algorithms. For nonlinear and multivariable mathematical models of mechanical systems, effective control methods have been recently developed within different approaches [1–3]. However, their implementation (with rare exceptions) requires measuring all variables of the state vector of the electromechanical system and acquiring current information about exogenous actions and their derivatives.

If some parameters of the plant and its environment, necessary to control the motion and orientation of the robot in the working space, cannot be directly measured, observers of the state and disturbances are used in the feedback loop [4–15]. These are algorithms implemented in a computing environment to restore the unknown signals from partial measurements of the state variables under the observability conditions. In tracking systems, it is rational to design an observer based not on the plant's original model but an equivalent canonical or block input–output form. Such an observer, whose dimension does not exceed that of the plant, combines several functions: observes the state and disturbance, identifies the parameters, and generates the given actions. By measuring tracking errors, it estimates mixed variables (functions of the state variables, the exogenous actions, and their derivatives) to form a feedback law [4, 7, 8, 14, 15]. As a result, the

[1]This work was supported by the Russian Foundation for Basic Research, project no. 20-01-00363-A.

controller's structure is simplified. In some cases, observers based on both the original and transformed models are used simultaneously for control purposes [13].

This paper considers a single-link manipulator elastically articulated with a DC motor in the case when the controlled angular position of the manipulator and its velocity cannot be qualitatively measured (due to an aggressive environment, sudden temperature changes or vibration [16]), and the sensors are installed only on the drive. For obtaining the tracking system under parametric and exogenous disturbances, we design a two-loop observer of the endogenous and exogenous variables. The observation subsystem structure proposed below differs from that used in [13], where the controlled variables were measured, and both observers were tuned independently of each other. In this paper, the first observer is based on the electrical subsystem and estimates its input (the controlled variable). Together with the command signal, this variable represents the output for the second observer. It is based on the virtual canonical input–output system and estimates the mixed variables to form a dynamic feedback law. Under uncertainty about the corrections of the second observer, an additive parasitic signal—an undamped estimation error of the first observer—appears.

The scientific novelty consists in designing an observation subsystem in the tracking system for a particular electromechanical plant without measuring the controlled variable. Also, a procedure for designing an observer with piecewise linear corrections for the canonical system with an uncertain input and an additive deterministic noise at the output is developed.

1. PLANT DESCRIPTION. THE BASIC CONTROL LAW

As the plant, we consider a single-link rigid manipulator with a rotating joint elastically connected to the shaft of a DC motor [17]:

$$\dot{x}_1 = x_2,\ \dot{x}_2 = a_{21}(x_3 - x_1) - a_{22}\sin(x_1) + f(t), \quad (1)$$

$$\dot{x}_3 = x_4,\ \dot{x}_4 = -a_{43}(x_3 - x_1) - a_{44}x_4 + a_{45}x_5,\ \dot{x}_5 = -a_{54}x_4 - a_{55}x_5 + bu. \quad (2)$$

Equations (1) describe the dynamics of the manipulator (the mechanical subsystem); equations (2), the dynamics of the DC motor with permanent magnets [3, 18] (the electrical subsystem). In addition, a_{ij} and b are positive design coefficients:

$$a_{21} = k_l / J_l,\ a_{22} = \overline{m}\overline{g}h / J_l,\ a_{43} = k_l / J_m,$$
$$a_{44} = d / J_m,\ a_{45} = k_m / J_m,$$
$$a_{54} = c / L,\ a_{55} = R / L,\ b = 1 / L.$$

The variables $x = (x_1, ..., x_5)^{\mathrm{T}}$ and the parameters of the system (1)–(2) are described in Table 1.

Table 1

The variables and parameters of the plant

Notation	Description, measurement unit
x_1	The angular position of plant's link, $\lvert x_1 \rvert < \pi$, rad
x_2	The angular velocity of plant's link, rad/s
$f(t)$	Uncontrolled disturbance, N/(kg·m)
x_3	The angular position of DC motor's shaft, rad
x_4	The angular velocity of DC motor's shaft, rad/s
x_5	The armature current of DC motor, A
u	The armature voltage of DC motor, V
$\overline{g} = 9.8$	Acceleration of gravity, m/s^2
$\overline{m}$	Manipulator's link mass, kg
h	Manipulator's link length, m
k_l	Gear rigidity, N·m/rad
J_l	The moment of inertia of manipulator's link, kg·m^2
d	Damping coefficient, kg·m^2/s
J_m	The moment of inertia of DC motor, kg·m^2
k_m	Gain, N·m/A
L	The armature inductance of DC motor, H
c	The counter emf coefficient of DC motor, V·s/rad
R	The armature resistance of DC motor, Ω

In the system (1) and (2), the output controlled variable is the angular position x_1 of the manipulator's link; the function $f(t)$ is interpreted as the uncontrolled exogenous disturbance; the control u is the armature voltage of the DC motor (from the class of discontinuous functions); $x \in X \subset R^5$, where X is an open bounded domain of variation of the state variables due to the existing design constraints.

For the system (1) and (2), the problem is designing a discontinuous feedback control under which the output variable $x_1(t)$ will track a given admissible signal $g(t)$ under the following assumptions:

– The sensors are located only on the actuator. The variables $x_3(t)$, $x_4(t)$,, and $x_5(t)$ are directly measured. The measurements contain no noise.

– The values of the parameters k_l, J_m, d, and k_m (hence, a_{43}, a_{44}, and a_{45}) are known. The parameters $\overline{m}$, h, J_l, c, R, and L (hence, a_{21}, a_{22}, a_{54}, a_{55}, and b) are uncertain but belong to given ranges.

– The time-varying function $f(t)$ is unknown, piecewise differentiable, and bounded, together with

its derivatives up to the third order inclusive, by given constants for $t \geq 0$.

– The reference signal $g(t)$ comes in real time from an autonomous source and is unknown in advance. The time-varying function $g(t)$ is assumed to be piecewise differentiable without any analytical description. In addition, its derivatives $g^{(i)}(t), i = \overline{1, 5}$, are unknown but bounded by given constants for $t \geq 0$.

– The generators of exogenous actions are not introduced into consideration, and the problem of estimating separately the derivatives of the reference signal and disturbance is not posed.

Under the assumptions made, the tracking error $e_1(t) = x_1(t) - g(t)$ can be stabilized only with some accuracy. Let $\Delta_1 > 0$ and $t^* > 0$ be the given stabilization accuracy and time to reach it. The goal of control is satisfying the following condition in the closed loop system:

$$|e_1(t)| \leq \Delta_1, t \geq t^*. \quad (3)$$

The system (1) and (2) is a special case of the Lagrangian completely driven system. For $f(t) \equiv 0$, it is controllable and observable in the output variable $x_1(t)$, and its relative order coincides with the system's dimension. Hence, it can be represented in the canonical input–output form without separating the endogenous dynamics [11, 14, 15]. As shown in [7], the canonical form of this class of systems is invariant with respect to exogenous affine disturbances. That is, the presence of an exogenous disturbance with a numerical factor in the second equation of the subsystem (1) will not change observability, controllability, and the relative order inherent in the undisturbed system when passing to the canonical basis of the mixed variables. Thus, differentiating five times the tracking error (3) and performing diffeomorphic changes of the state variables with an affine entry of the exogenous actions and their derivatives, we represent the plant's mathematical model (1) and (2) in the canonical form

$$\dot{e}_i = e_{i+1}, i = \overline{1,4}, \quad (4)$$

$$\dot{e}_5 = \psi(e, t) + \tilde{b}u \quad (5)$$

in the mixed variables $e = (e_1, e_2, e_3, e_4, e_5)^{\mathrm{T}}$:

$$\begin{gathered} e_2 = x_2 - \dot{g}(t), \\ e_3 = a_{21}(x_3 - x_1) - a_{22}\sin(x_1) + f(t) - \ddot{g}(t), \\ e_4 = a_{21}(x_4 - x_2) - a_{22}x_2\cos(x_1) + \dot{f}(t) - \dddot{g}(t), \\ e_5 = a_{21}[(a_{43} + a_{21} + a_{22}\cos(x_1))(x_1 - x_3) - \\ -a_{44}x_4 + a_{45}x_5] + (a_{21} + a_{22}\cos(x_1) + x_2^2)a_{22}\sin(x_1) - \\ -(a_{21} + a_{22}\cos(x_1))f(t) + \ddot{f}(t) - g^{(4)}(t), \end{gathered} \quad (6)$$

where $\tilde{b} = a_{21}a_{45}b > 0$, and the function $\psi(e, g, \dot{g}, ..., g^{(5)}, f, \dot{f}, \ddot{f}, \dddot{f}) = \frac{d}{dt}e_5 - \tilde{b}u$, obtained after the inverse change of the variables (6), is treated as a bounded disturbance. Using the known ranges of the external actions, their derivatives, and the design constraints on the variables and parameters of the system (1) and (2), we arrive at the bounds

$$|e_i(0)| \leq E_{0i}, i = \overline{1,5}; \quad \underline{b} \leq \tilde{b} \leq \overline{\overline{b}}. \quad (7)$$

We design a discontinuous control for obtaining a sliding mode in the closed loop system on the surface $s = c_1e_1 + c_2e_2 + c_3e_3 + c_4e_4 + e_5 = 0$ in the virtual space of the canonical variables $e \in R^5$ (6), where $c_i = \mathrm{const} > 0$ are the coefficients of the Hurwitz polynomial $\prod_{i=1}^{4}(\lambda - \lambda_i) = \lambda^4 + c_4\lambda^3 + c_3\lambda^2 + c_2\lambda + c_1, \mathrm{Re}\lambda_i < 0,$ $i = \overline{1,4}$, assigned according to the existing requirements to the tracking error transients. For the system (4) and (5), the basic discontinuous control law with a constant amplitude $M = \mathrm{const} > 0$ has the form

$$u = -M\mathrm{sign}(s), \quad (8)$$

where

$$\mathrm{sign}(s) = \begin{cases} +1, & s > 0, \\ -1, & s < 0. \end{cases}$$

(For $s = 0$, the value of this function is indefinite but belongs to the interval $[-1, 1]$.)

The lower bound on the amplitude is found from the sufficient condition $s\dot{s} < 0$ (see [3]):

$$\dot{s} = c_1e_2 + c_2e_3 + c_3e_4 + c_4e_5 + \psi(e, t) + \tilde{b}u,$$

$$s\dot{s} \leq |s|\left(\sum_{i=1}^{4} c_i|e_{i+1}| + |\psi(e, t)| - \underline{b}M\right).$$

It holds under

$$M > \left(\sum_{i=1}^{4} c_i|e_{i+1}| + |\psi(e, t)|\right)/\underline{b} \quad (9)$$

and the desired time $t_s : 0 < t_s < t^*$ of sliding mode occurrence:

$$M \geq \frac{1}{\underline{b}}\left(\frac{|s(0)|}{t_s} + \sum_{i=1}^{4} c_i|e_{i+1}| + |\psi(e, t)|\right),$$

$$|s(0)| \leq \sum_{i=1}^{4} c_iE_{0i} + E_{05} = S_0. \quad (10)$$

Under the control law (8) and (10), the function $s(t)$ will monotonically vanish in a finite time in the closed loop virtual system [3] if the prior assumptions used for deriving the initial conditions (7) hold along the entire closed loop system trajectory.

Due to $e_5 = s - (c_1 e_1 + c_2 e_2 + c_3 e_3 + \ + c_4 e_4)$, we obtain the stable reduced system

$$\dot{e}_i = e_{i+1},\ i = \overline{1,3};\ \dot{e}_4 = -(c_1 e_1 + \\ + c_2 e_2 + c_3 e_3 + c_4 e_4) + s, \quad (11)$$

where $s(t) = 0,\ t \ge t_s$, and consequently, the asymptotic stabilization of the tracking error:

$$\lim_{t\to\infty} e_1(t) = 0 \Leftrightarrow \lim_{t\to\infty} x_1(t) = g(t).$$

The ranges of the mixed variables necessary for (10) are found using (11), (7), $|s(t)| \le S_0,\ t \ge 0$, and the accepted values $c_i,\ i = \overline{1,4}$:

$$|e_i(t)| \le E_i,\ i = \overline{1,4};\ |\psi(e,t)| \le \Psi, t \ge 0; \\ |e_5(t)| \le S_0 + (c_1 E_1 + c_2 E_2 + \\ + c_3 E_3 + c_4 E_4) = E_5. \quad (12)$$

The system (11) describes motion in an ideal sliding mode: the representation point $e(t) \in R^5$ in a finite time t_s falls on the switching surface $s = 0$ and moves along it to an equilibrium for $t > t_s$, theoretically oscillating with an infinitesimal amplitude and infinite frequency [3].

The feedback law (8) is formed by the mixed variables $e_i(t), i = \overline{1,5}$ (6). The problem is estimating them using a dynamic observer by the measured signals $g(t),\ x_3(t),\ x_4(t),$ and $x_5(t)$. Under parametric uncertainty and no dynamic generators of the exogenous actions, special approaches are needed for designing and tuning the observation subsystem; see Section 2. Note that under these conditions, the observation problem can be solved only with some accuracy. Let $\tilde{e}_i(t)$ be the mixed variables estimates yielded by the observation subsystem, and let $\delta_i > 0$ and $T > 0$ be the corresponding accuracy and time of solving the observation problem. Under the conditions

$$|e_i(t) - \tilde{e}_i(t)| \le \delta_i,\ i = \overline{1,5},\ t \ge T,\ T < t_s, \quad (13)$$

a real sliding mode will occur in the closed loop virtual system (4)–(5) with the dynamic feedback law

$$u = -M\text{sign}(\tilde{s}),\ \tilde{s} = c_1\tilde{e}_1 + c_2\tilde{e}_2 + c_3\tilde{e}_3 + c_4\tilde{e}_4 + \tilde{e}_5, \quad (14)$$

in a finite time $t_s \ge T$ [3]. That is, for $t \ge t_s$, the representation point of the system (11) will move in the boundary layer of the switching surface $|s(t)| \le \Delta \le (c_1\delta_1 + c_2\delta_2 + c_3\delta_3 + c_4\delta_4 + \delta_5)$, which reduces the control accuracy in the steady-state mode. Given the gains $c_i,\ i = \overline{1,4}$, the admissible range $\Delta \le \bar{\Delta}$ corresponding to the desired tracking error stabilization accuracy Δ_1 (3) and the constraints (13) on the estimation accuracy $\delta_i,\ i = \overline{1,5}$, in the observation subsystem, are found from the condition

$$(c_1\delta_1 + c_2\delta_2 + c_3\delta_3 + c_4\delta_4 + \delta_5) \le \bar{\Delta}. \quad (15)$$

For sufficiently small observation errors (13), the equality $\text{sign}(\tilde{s}) = \text{sign}(s)$ holds beyond the range $|s(t)| \le \Delta$. Due to the mismatching initial conditions of the observer and observed system, the difference between the variables $e_i(t)$ and their estimates $\tilde{e}_i(t)$ (hence, between $s(t)$ and $\tilde{s}(t)$) is significant at the beginning of the transients. Therefore, in the system with the dynamic feedback law (14), the sufficient conditions for sliding mode occurrence (9) may be violated for $t \in [0, T]$; in the worst case, the monotonic convergence of $s(t)$ to the Δ-neighborhood of zero is guaranteed only for $t \ge T$. Due to the inequality $|s(T)| \le |s(0)| + (\sum_{i=1}^{4} c_i |e_{i+1}| + |\psi(e,t)| + M\bar{\bar{b}})T$, we refine the lower bound (10):

$$M \ge \frac{1}{\bar{\bar{b}}}\left(\frac{|s(T)| - \Delta}{t_s - T} + \sum_{i=1}^{4} c_i |e_{i+1}| + |\psi(e,t)|\right) \Rightarrow$$

$$\Rightarrow M \ge \frac{|s(0)| - \Delta + (\sum_{i=1}^{4} c_i |e_{i+1}| + |\psi(e,t)|)t_s}{\bar{b}t_s - (\bar{b} + \bar{\bar{b}})T}. \quad (16)$$

Under a fixed value $t_s < t^*$, the expression (16) yields the following constraint on the estimation time (13):

$$0 < T < \bar{b}t_s / (\bar{b} + \bar{\bar{b}}). \quad (17)$$

This constraint must be satisfied in the observation subsystem.

2. DOUBLE-LOOP OBSERVER DESIGN

2.1. Observer for controlled variable

The first loop estimates the manipulator's angular position by measuring the state variables of the electrical subsystem. Together with the reference signal, the estimate will be used to design the observer in the second loop. It estimates the mixed variables, necessary to form the feedback law (14), and is based on the virtual input–output system (4) and (5). In this system, all functional and exogenous actions are concentrated in equation (5) and do not narrow the observable space of the canonical variables with respect to the tracking error. Such a design of the observation subsystem is convenient for tracking systems: implementing the reverse changes of the variables in real time is not required; under uncertainty, there is no need to estimate additionally the exogenous actions and their derivatives and identify the parameters.

For $f(t) \equiv 0$, the system (1) and (2) is observable in the measurements $x_3(t),\ x_4(t),$ and $x_5(t)$. However,

the presence of parametric and exogenous disturbances in the mechanical subsystem (1) is an obstacle to constructing a full-order state observer. To estimate the unmeasured controlled variable $x_1(t)$, we will employ the ideology of estimating exogenous signals by their effect on the plant using "force" corrections in the form of deep feedbacks, discontinuous controls, or their combinations. In this case, the disturbance observer is based on some part of the plant's dynamic model subjected to an exogenous bounded signal. (The other variables and parameters are known [5, 6].) Within this approach, it suffices to know only the ranges of an exogenous signal under estimation, whereas its dynamic model is not used for observer design.

Since the values of the parameters a_{43}, a_{44} and a_{45} are known, we can design a reduced observer using the second equation of the electrical subsystem (2): its right-hand side is a linear combination of known signals and the variable $x_1(t)$ under estimation. Thus, the dynamic observer for estimating the manipulator's angular position (differentiator) will have the first order:

$$\dot{z} = -a_{43}x_3 - a_{44}x_4 + a_{45}x_5 + v, \qquad (18)$$

where z denotes the state variable, and v is the observer's correction. Due to the expressions (2) and (18), the observation error $\varepsilon = x_4 - z$ satisfies the equation

$$\dot{\varepsilon} = a_{43}x_1 - v, \qquad (19)$$

where the unknown signal $x_1(t)$ is treated as an exogenous disturbance bounded by a given constant: $|x_1(t)| < \pi,\ t \ge 0$. Choosing an appropriate action $v(\varepsilon)$, we have to stabilize in a given time with a desired accuracy the observation error and its derivative:

$$|\varepsilon(t)| \le \alpha,\ |a_{43}x_1(t) - v(t)| \le \overline{\alpha},\ t \ge T_1. \qquad (20)$$

Then, for sufficiently small $\overline{\alpha} > 0$ and $t \ge T_1$, the correction $v(t)$ can be adopted to estimate the controlled variable.

We will solve the problem (20) using an observer with piecewise linear feedback [6, 7, 13], a hybrid of discontinuous and linear controls with high gains:

$$v = m\,\mathrm{sat}(l\varepsilon) = \begin{cases} m\,\mathrm{sign}(\varepsilon), & |\varepsilon| > 1/l, \\ ml\varepsilon, & |\varepsilon| \le 1/l,\ m, l = \mathrm{const} > 0. \end{cases} \qquad (21)$$

It has two tunable parameters: the amplitude $m = \mathrm{const} > 0$, determining the time of convergence of the observation error to the neighborhood of zero, where the correction (21) is linear; a high gain $l = \mathrm{const} > 0$, determining in inverse proportion the radius of this neighborhood (hence, the estimation accuracy).

Beyond the linear zone, the system (19) and (21) has the form $\dot{\varepsilon} = a_{43}x_1 - m\,\mathrm{sign}(\varepsilon)$. By analogy with discontinuous control systems operating in a sliding mode (9), we find a lower bound on the amplitude. The sufficient condition $\varepsilon\dot{\varepsilon} < 0$, where $\varepsilon\dot{\varepsilon} = \varepsilon(a_{43}x_1 - m\,\mathrm{sign}(\varepsilon)) \le |\varepsilon|(a_{43}\pi - m)$, holds under

$$m > a_{43}\pi, \qquad (22)$$

which will ensure the inequality $|\varepsilon(t)| \le 1/l$ in a finite time.

Because the variable $x_4(t)$ is measured, we can guarantee zero initial conditions for the observation error: $z(0) = x_4(0) \Rightarrow \varepsilon(0) = 0$. Then the first inequality of (21) will hold for all $t \ge 0$ under any amplitude (22) and $l \ge 1/\alpha$.

In the linear zone, the system (19) and (21) has the form $\dot{\varepsilon} = a_{43}x_1 - ml\varepsilon$. To ensure the second inequality of (20), we estimate the system's solution for $t \in [0, T_1]$:

$$|\varepsilon(T_1)| \le \frac{a_{43}|x_1(t)|}{ml} + \left(\frac{1}{l} - \frac{a_{43}|x_1(t)|}{ml}\right)e^{-mlT_1} \le$$

$$\le \frac{a_{43}\pi}{ml} + \frac{m - a_{43}\pi}{ml}e^{-mlT_1},$$

$$ml|\varepsilon(T_1)| - a_{43}\pi \le (m - a_{43}\pi)e^{-mlT_1},$$

$$|a_{43}x_1(t) - v(t)| \le \overline{\alpha},\ t \ge T_1 \Leftrightarrow (m - a_{43}\pi)e^{-mlT_1} \le \overline{\alpha}. \qquad (23)$$

From (23) it follows that for $t \ge T_1$, the observation error will converge to the neighborhood of zero:

$$|\varepsilon(t)| \le \frac{a_{43}\pi + \overline{\alpha}}{ml}.$$

Since $m > a_{43}X_1$ (22), the lower bound $l \ge 1/\alpha$ can be improved: both inequalities of (20) will hold under the high gain

$$l \ge \frac{1}{m}\max\left\{\frac{a_{43}\pi + \overline{\alpha}}{\alpha};\ \frac{1}{T_1}\ln\frac{m - a_{43}\pi}{\overline{\alpha}}\right\}.$$

Thus, for $t > T_1$, the controlled variable $x_1(t)$ is estimated by the signal

$$\widetilde{x}_1(t) = v(t)/a_{43} :\ |x_1(t) - \widetilde{x}_1(t)| = |\eta_1(t)| \le \overline{\alpha}/a_{43} = \alpha_1. \qquad (24)$$

Together with the reference signal $g(t)$, it comes to the second loop and serves to tune the observer of the mixed variables $e_i(t),\ i = \overline{1, 5}$, (6) of the canonical system (4) and (5). As a result, the accuracy and time of estimating the mixed variables in the second loop depend on the accuracy and time of estimating the controlled variable in the first loop and

$$\overline{\alpha}/a_{43} = \alpha_1 < \delta_1,\ T_1 < T. \qquad (25)$$

Recall that the upper constraints (15) and (17) have been imposed on inequalities (25), respectively, and a desired estimation accuracy can be provided in the first loop (18) in a given time. Considering these features, we will specify the formal requirements to $\alpha_1(\delta_1)$ and $T_1(T)$ (25) when tuning the second loop.

2.2. Reduced-order observer for mixed variables

It is difficult to design a full-order observer in the second loop since the factor $\tilde{b} = a_{21}a_{45}b$ at the discontinuous control is not known exactly. Due to the impossibility of fully compensating the control signal in the system with respect to observation errors, a high-frequency parasitic signal will be added to the existing uncertainties $\psi(e, t)$. As a result, the quality of the estimation may worsen. To avoid this problem, we will estimate the mixed variables using a reduced-order observer based on the linear subsystem (4) in the form

$$\dot{z}_i = z_{i+1} + v_i,\ i = \overline{1,3};\ \dot{z}_4 = v_4, \qquad (26)$$

where z_i denote the observer's state variables, and v_i are its corrections. According to the expressions (4) and (26), the observation errors $\varepsilon_i = e_i - z_i,\ i = \overline{1,4}$, satisfy

$$\dot{\varepsilon}_i = \varepsilon_{i+1} - v_i,\ i = \overline{1,3};\ \dot{\varepsilon}_4 = e_5 - v_4, \qquad (27)$$

where $e_5(t)$ is treated as an exogenous bounded disturbance such that $|e_5(t)| \le E_5,\ t \ge 0$ (12). The mixed variables estimates $\tilde{e}_i(t),\ i = \overline{1,4}$, will be obtained using the observer's variables $\tilde{e}_i(t) = z_i(t)$, and by analogy with the system (19), the exogenous disturbance will be estimated by the correction $\tilde{e}_5(t) = v_4(t)$. In terms of the system (27), the stabilization problem (13) with the constraints (15) and (17) takes the form

$$|\varepsilon_i(t)| = |e_i(t) - z_i(t)| \le \delta_i,\ i = \overline{1,4},$$
$$|e_5(t) - v_4(t)| \le \delta_5,\ t \ge T. \qquad (28)$$

To solve the problem, we will also use piecewise linear corrections formed according to the cascade principle [7]:

$$v_i = m_i \mathrm{sat}(l_i v_{i-1}) = \begin{cases} m_i \mathrm{sign}(v_{i-1}), & |v_{i-1}| > 1/l_i, \\ m_i l_i v_{i-1}, & |v_{i-1}| \le 1/l_i, \end{cases} \quad i = \overline{1,4}, \qquad (29)$$

where $v_0(t) = \tilde{x}_1(t) - g(t) - z_1(t) = \varepsilon_1(t) - \eta_1(t)$. The observer (26) and (29) differs from the one presented in [7]: the variable $v_0(t)$ used to form the feedback law (29) additively contains an undamped parasitic signal $\eta_1(t)$, which cannot be compensated. However, for $t > T_1$, it can be made arbitrarily small in absolute value.

Setting zero initial conditions in the observer (26), $z_i(0) = 0 \Rightarrow \varepsilon_i(0) = e_i(0),\ i = \overline{1,4}$, we take the estimates (7) as the initial conditions domain for the observation errors:

$$|\varepsilon_i(0)| = |e_i(0)| \le E_{0i},\ i = \overline{1,4}. \qquad (30)$$

The following result expresses sufficient conditions for solving problem (28).

Lemma. Let the initial conditions and the function $e_5(t)$ in the system (27) and (29) be bounded by known constants (30) and (12), respectively. Then for any $T > 0$ and $\delta > 0$, there exist real numbers $m_i^*, l_i^* > 0$ such that inequalities (28) hold for all m_i, l_i: $m_i \ge m_i^*,\ l_i \ge l_i^*,\ i = \overline{1,4}$.

A constructive proof of the lemma is presented in the Appendix. In this proof, we formalize an iterative procedure for tuning the parameters of the corrections (29) with the separation of motions in the observation errors space. This procedure can be easily extended to higher-order canonical systems and square canonical systems with vector variables. Also, we establish requirements to the estimation time and accuracy (25) in the first loop of the observation subsystem in the form

$$0 < T_1 \le T/46,\ \alpha_1 = \delta_1/2 \Rightarrow 0 < \bar{\alpha} \le a_{43}\delta_1/2.$$

Note that the tracking and observation errors are stabilized using sufficient stability conditions, which yield conservative (overestimated) lower bounds on the dynamic feedback parameters. They can be refined based on the simulation results.

Within the developed approach, the corrections in the observers and the discontinuous control law are both bounded by absolute value. Therefore, the well-known problem of a large overshoot at the beginning of the transients, arising in closed loop systems with linear feedback by the variables of linear observers with high gains, is eliminated here. The problem is considered in a deterministic statement. The applicability of observers with piecewise linear corrections under an uncertain input and noisy measurements requires a separate study, going beyond the scope of this paper.

3. SIMULATION RESULTS

The developed algorithms were numerically simulated in MATLAB-Simulink for the system (1) and (2) with the initial conditions $x(0) = \bar{0}$. Integration was performed using Euler's method with a constant step of 10^{-5}. For ensuring the target values $\Delta_1 = 0.05$ rad and $t^* = 3$ s of the control performance indices under condition (3), the following values of the feedback parameters (14) were adopted:

$$c_1 = 625,\ c_2 = 500,\ c_3 = 150,\ c_4 = 20,\ M = 90. \qquad (31)$$

The worst case was considered with the admissible values of the design coefficients and exogenous actions given by

$$k_l = 0.2,\ J_m = 0.01,\ d = 0.045,\ k_m = 0.3;$$

$\overline{m} \in [0.2, 0.8]$, $h \in [0.15, 1.2]$, $J_l \in [0.045, 1.15]$, $c \in [0.25, 0.33]$,

$R \in [3.8, 4.2]$, $L \in [0.006, 0.013]$;

$\left|g^{(i)}(t)\right| < \pi$, $i = \overline{0,5}$, $\left|f^{(i)}(t)\right| \le 0.01$, $i = \overline{0,3}$, $t \ge 0$.

The parameters of the piecewise linear corrections (21) and (29) of the observers (18) and (26), respectively, and the initial conditions were chosen as

$$m = 280,\ l = 30,\ z(0) = x_4(0); \qquad (32)$$

$$m_1 = 150,\ m_2 = 116,\ m_3 = 150,\ m_4 = 100,\ l_1 = 30,$$

$$l_2 = 30,\ l_3 = 10,\ l_4 = 2,\ z_i(0) = 0,\ i = \overline{1,4}.$$

Two numerical experiments with the same feedback parameters (31) and (32) and different uncertain coefficients and exogenous actions from the indicated ranges were carried out. In each experiment, three setups were examined:

- the reference system (1) and (2) with the basic control law (8) under complete information about the system coefficients, endogenous and exogenous signals, and their derivatives;
- the system (1) and (2) with the sensorless drive and measurements of the tracking error $e_1(t) = x_1(t) - g(t)$ only (the mixed variables $e_i(t)$, $i = \overline{2,5}$, were estimated using the observer (26) and (29); the correction $v_0(t) = e_1(t) - z_1(t) = \varepsilon_1(t)$ contained no parasitic signal $\eta_1(t)$; the values k_l, J_m, d, and k_m were uncertain; the control law (14) was based on $\tilde{s} = c_1 e_1 + c_2 z_2 + c_3 z_3 + c_4 z_4 + v_4$);
- the system (1) and (2) with the sensorless manipulator and measurements of the parameters $g(t)$, $x_3(t)$, $x_4(t)$, and $x_5(t)$ (the double-loop observers (18), (21) and (26), (29) were applied, and the control law (14) was implemented using the estimate (24) based on $\tilde{s} = c_1(v/a_{43} - g) + c_2 z_2 + c_3 z_3 + c_4 z_4 + v_4$).

Experiment 1: $\overline{m} = 0.5$, $h = 1$, $J_l = 0.5$, $c = 0.2865$, $R = 4.1$, $L = 0.0103$;

$$f(t) = 0.01\sin(t);$$

$$g(t) = \begin{cases} +\pi/3,\ t \in [0, 5], [10, 15], [20, 25], \ldots; \\ -\pi/3,\ t \in (5, 10), (15, 20), (25, 30), \ldots \end{cases}$$

For the reference system, Fig. 1 shows the graphs of the reference signal $g(t)$ and the manipulator's angular position $x_1(t)$; Fig. 2, the graph of the tracking error $e_{1\text{ref}}(t) = x_1(t) - g(t)$ (the dashed line indicates the admissible error range, $|e_{1\text{ref}}(t)| \le \Delta_1 = 0.05$ rad, $t \ge t^* = 3$ s, in the steady-state mode). Figure 3 shows the graphs of $(e_{1\text{ref}}(t) - e_{1\text{dr}}(t))$ and $(e_{1\text{ref}}(t) - e_{1\text{man}}(t))$, representing the deviations from the tracking errors $e_{1\text{dr}}(t)$ (the system with the sensorless drive) and $e_{1\text{man}}(t)$ (the system with the sensorless manipulator). The dashed line indicates the admissible error range. For the system with the sensorless manipulator, Fig. 4 shows the graph of $x_1(t) - \tilde{x}_1(t)$, representing the estimation error of the angular position $x_1(t)$ using the observer (18) with $\tilde{x}_1(t) = v(t)/a_{43}$.

Experiment 2: $\overline{m} = 0.45$, $h = 1.1$, $J_l = 0.5445$, $c = 0.25$, $R = 3.9$, $L = 0.0103$;

$$g(t) = \frac{\pi}{3}\sin(0.5t);$$

$$f(t) = \begin{cases} -0.01,\ t \in [0, 2], [4, 6], [8, 10], \ldots; \\ +0.01,\ t \in (2, 4), (6, 8), (10, 12), \ldots \end{cases}$$

Figures 5–8 show graphs similar to the ones in Figs. 1–4.

For both experiments and all setups, Table 2 presents the values of the control performance indices: the settling time $t^*: |e_1(t)| \le 0.05,\ t \ge t^*$; the overshoot $e_{1\max} \ge |e_1(t)|$, and the tracking accuracy $\overline{\Delta}_1 \ge |x_1(t) - g(t)|$ in the steady-state mode.

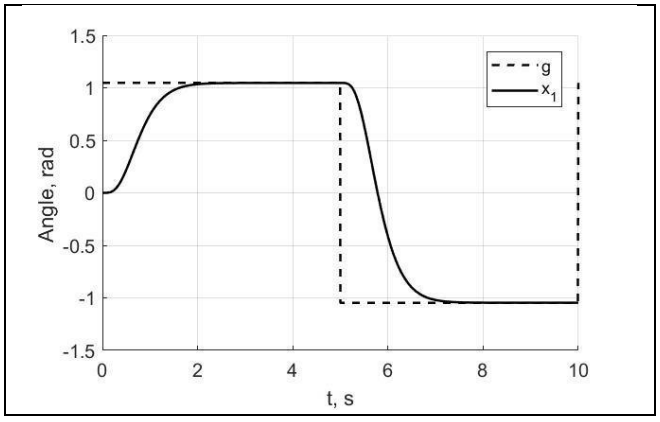


Fig. 1. Graphs of parameters $g(t)$ and $x_1(t)$ for reference system.

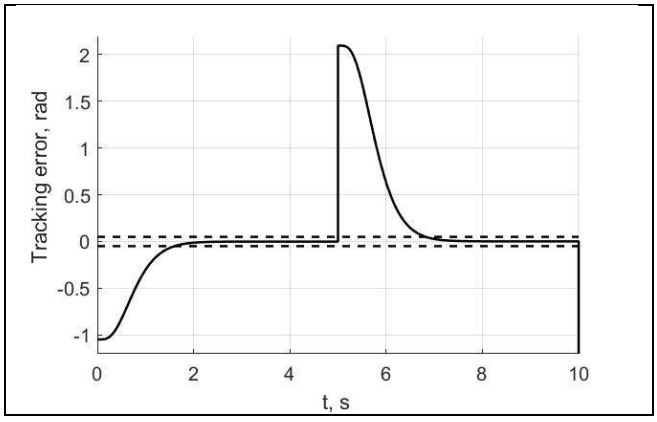


Fig. 2. Graph of tracking error $e_{1\text{ref}}(t) = x_1(t) - g(t)$ for reference system.

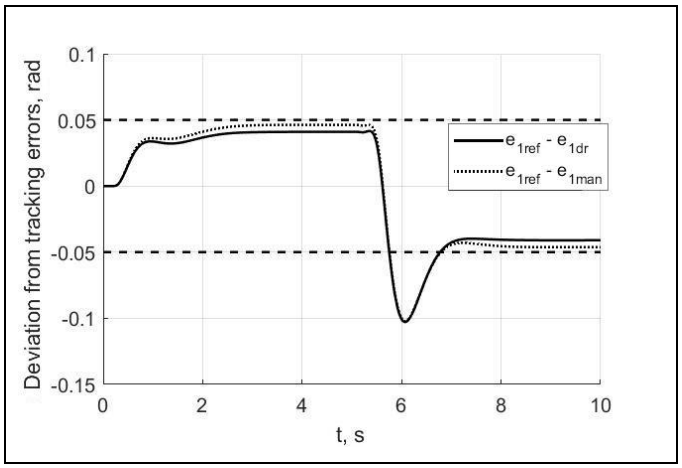


Fig. 3. Graphs of deviations from tracking errors, $e_{1ref}(t) - e_{1dr}(t)$ and $e_{1ref}(t) - e_{1man}(t)$, for reference system.

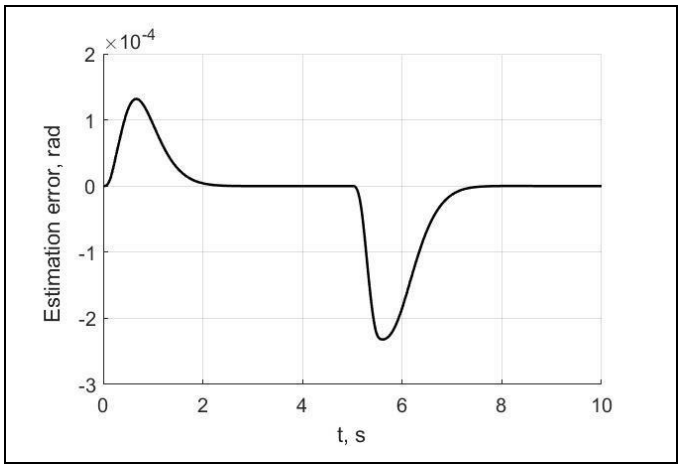


Fig. 4. Graph of estimation error $x_1(t) - \tilde{x}_1(t)$.

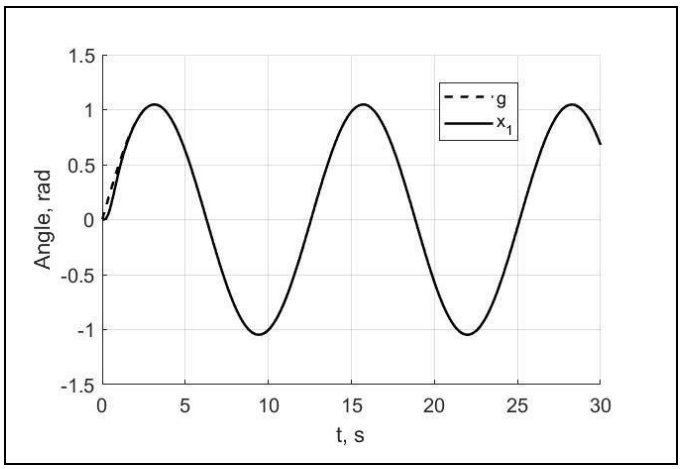


Fig. 5. Graphs of parameters $g(t)$ and $x_1(t)$ for reference system.

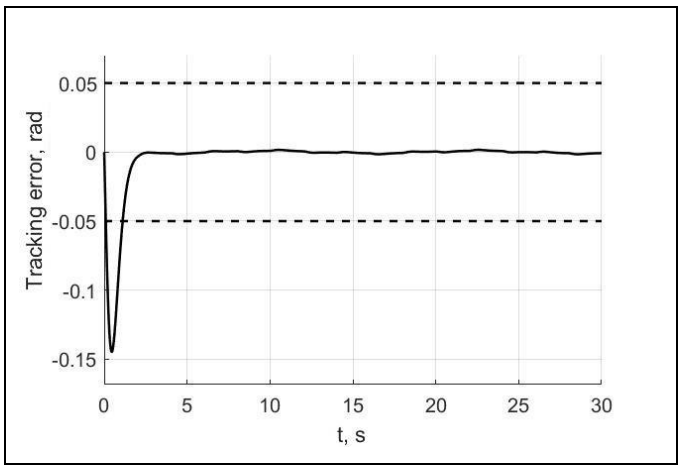


Fig. 6. Graph of tracking error $e_{1ref}(t) = x_1(t) - g(t)$ for reference system.

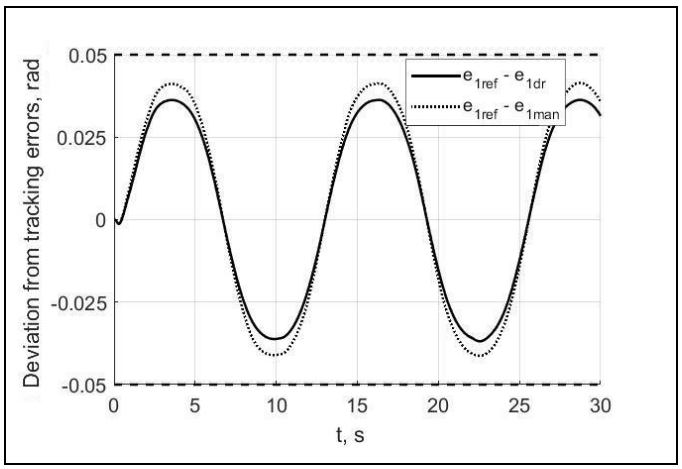


Fig. 7. Graphs of deviations from tracking errors, $e_{1ref}(t) - e_{1dr}(t)$ and $e_{1ref}(t) - e_{1man}(t)$, for reference system.

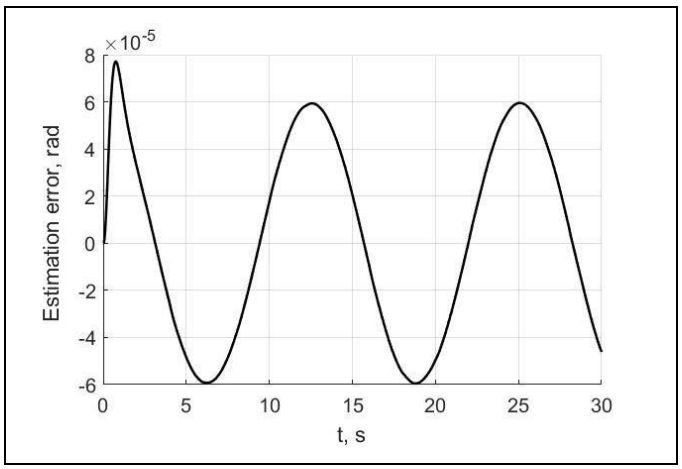


Fig. 8. Graph of estimation error $x_1(t) - \tilde{x}_1(t)$.

Table 2

Values of control performance indices

Performance index, measurement unit	Reference system under complete information	System with sensorless drive	System with sensorless manipulator
Experiment 1 (for the first transient, $0 \le t \le 5$)			
t^*, s	1.6039	1.9968	2.1753
$e_{1\max}$, rad	1.0472	1.0472	1.0472
$\overline{\Delta}_1$, rad	0.0013	0.0421	0.0473
Experiment 2			
t^*, s	1.1137	1.2292	1.2568
$e_{1\max}$, rad	0.1448	0.1448	0.1449
$\overline{\Delta}_1$, rad	0.0016	0.0385	0.0430

According to Table 2, the target values of the control performance indices are achieved in all cases. In addition, the same overshoot of the tracking error is provided in the reference system and the systems with observers. In the systems with an incomplete set of sensors, the settling time increases insignificantly; the stabilization accuracy of the tracking error in the steady-state mode worsens by about 30 times compared to the reference system with an almost ideal sliding mode. (This worsening is due to the observation errors that generate a real sliding mode in the boundary layer of the switching surface $|s(t)| \le \Delta$.) However, comparison with the reference system is not constructive: it cannot be implemented in practice since the basic control law (8) requires accurate knowledge about the plant's parameters and, moreover, the exogenous disturbances and their derivatives. It is of interest to compare the systems with incomplete measurements.

In the steady-state mode, the settling time and accuracy in the system with the sensorless manipulator are worse than, albeit commensurable with, those in the system with the sensorless drive. The reason is a fast convergence of the controlled variable observer (see Figs. 4 and 8) due to the zero initial observation error in the system (19). Thus, in the closed loop systems with an incomplete set of sensors and dynamic feedback laws, various admissible scenarios are observed when varying the parameters of the plant and exogenous factors within the calculated ranges without controller retuning.

CONCLUSIONS

For an electromechanical system with a single-link manipulator, a double-loop observer and a decomposition procedure for designing linear corrections with saturation have been developed in the case when the output variable contains an uncompensated parasitic signal. The total dimension of the observation subsystem, restoring all endogenous and exogenous signals necessary for feedback control design, is equal to the plant's dimension.

The results of numerical simulation have confirmed the efficiency of the developed dynamic feedback control design method. As shown, it suffices to know the ranges of variation for the parameters of the plant, exogenous disturbances, and their derivatives when designing feedback control based on the mixed variables in the observer's tracking system. Under uncertainty (without the controlled variable sensor and the generator of exogenous actions), the tracking problem is solved with a desired accuracy in a given time.

Without loss of generality, the new design method of double-loop observation subsystems can be extended to completely driven electromechanical tracking systems with multi-link sensorless manipulators, which are described by a system of differential equations of the same order, similar to the system (1) and (2), in the vector variables with matrix coefficients [8].

APPENDIX

P r o o f of the lemma. Denoting $0 = t_0 < T_1 = t_1 < t_2 < t_3 < \ldots < t_8 < t_9 = T$, we formalize the desired dynamics of the observation errors in the closed loop system (27) and (29) according to the target (28):

$$|v_{i-1}(t)| \le 1/l_i \Leftrightarrow |\varepsilon_i(t)| \le \alpha_i + 1/l_i,\ t \ge t_{2i}; \quad \text{(A.1)}$$

$$|\varepsilon_{i+1}(t) - v_i(t)| = |\eta_{i+1}(t)| \le \alpha_{i+1} < \delta_{i+1},$$
$$|\varepsilon_i(t)| \le \delta_i,\ t \ge t_{2i+1},\ i = \overline{1,\ 4}. \quad \text{(A.2)}$$

Inequalities (A.1) are ensured by choosing appropriate amplitudes $m_i > 0,\ i = \overline{1,\ 4},$ and the corresponding corrections; inequalities (A.2), by choosing appropriate high gains $l_i > 0,\ i = \overline{1,\ 4}.$

In the system (27) and (29), the sign identity $\operatorname{sign}(v_i(t)) = \operatorname{sign}(\varepsilon_i(t)),\ i = \overline{1,\ 4},$ may not hold for $0 \le t \le t_{2i-1}$, being guaranteed only for $t > t_{2i-1}$ beyond

the neighborhood $|\varepsilon_i| \le \alpha_i$ (A.1).

In the general case $\delta_i << E_{01}$, $i = \overline{1, 4}$, due to (A.1) and (A.2), the system (27) and (29) can be written as

$$\dot{\varepsilon}_i = \begin{cases} \varepsilon_{i+1} + m_i \text{sign}(\varepsilon_i),\ t \in [t_0, t_{2i-1}), \\ \varepsilon_{i+1} - m_i \text{sign}(\varepsilon_i),\ t \in [t_{2i-1}, t_{2i}), \\ \varepsilon_{i+1} - m_i l_i (\varepsilon_i \pm \eta_i),\ t \ge t_{2i};\ i = \overline{1, 4},\ \varepsilon_5 := e_5. \end{cases} \quad (A.3)$$

Using (A.2), we estimate the variations of the observation errors in the closed loop system (A.3) as follows:

$$|\varepsilon_i(t)| \le |\varepsilon_i(t_{2i-1})| \le E_{0i} + (F_{i+1} + m_i) t_{2i-1} = \\ = F_i,\ i = \overline{1, 4}, F_5 = E_5. \quad (A.4)$$

By analogy with the expressions (10) and (11) (from the sufficient conditions $\varepsilon_i \dot{\varepsilon}_i < 0$ on the indicated time intervals (A.3)), we construct hierarchical inequalities for the amplitudes under which, successively top-to-bottom, the correction arguments will converge to the linear zones (A.1):

$$m_i \ge \frac{|\varepsilon_i(t_{2i-1})|}{t_{2i} - t_{2i-1}} + F_{i+1},\ i = \overline{1, 4}.$$

Substituting the estimates (A.4) into the resulting inequalities, we obtain a sequential (bottom-to-top) hierarchical scheme for choosing the correction amplitudes:

$$m_i \ge \frac{E_{0i} + (F_{i+1} + m_i) t_{2i-1}}{t_{2i} - t_{2i-1}} + F_{i+1} \Rightarrow \\ \Rightarrow m_i^* = \frac{E_{0i} + F_{i+1} t_{2i}}{t_{2i} - 2t_{2i-1}},\ i = \overline{4, 1}. \quad (A.5)$$

The scheme (A.5) implies the time constraints $t_{2i} > 2t_{2i-1}$, $i = \overline{1, 4}$. For example, let T_1 be a time increment:

$$t_1 = T_1,\ t_{2i} = 2t_{i-1} + T_1,\ t_{2i+1} = t_{2i} + T_1,\ i = \overline{1, 4}. \quad (A.6)$$

Given the value T (17), the convergence time (20) of the controlled variable observer (18) will satisfy the following upper bound:

$$T = [3(1 + 2 + 2^2 + 2^3) + 1] T_1 \Rightarrow 0 < T_1 \le T / 46. \quad (A.7)$$

Substituting the expressions (A.6) and (A.7) into the scheme (A.5) yields the values m_i^*: $\forall m_i \ge m_i^*$, $i = \overline{1, 4}$, and inequalities (A.1) will hold.

To determine the lower bounds on the high gains $l_i > 0$ ensuring inequalities (A.2), similar to the expression (23), we consider the estimates of the *i*th variables described by the third equations of (A.3) on the time intervals $[t_{2i}, t_{2i} + T_1 = t_{2i+1}]$, $i = \overline{1, 4}$:

$$|\varepsilon_i(t_{2i+1})| \le \frac{|\varepsilon_{i+1}(t)|}{m_i l_i} + |\eta_i(t)| + \left(\frac{1}{l_i} - \frac{|\varepsilon_i(t)|}{m_i l_i} \right) e^{-m_i l_i T_1} \le \\ \le \frac{F_{i+1}}{m_i l_i} + \alpha_i + \frac{m_i - F_{i+1}}{m_i l_i} e^{-m_i l_i T_1}, \\ m_i l_i (|\varepsilon_i(t_{2i+1})| - \alpha_i) - F_{i+1} \le (m_i - F_{i+1}) e^{-m_i l_i T_1}, \quad (A.8) \\ |\varepsilon_{i+1}(t) - v_i(t)| \le \alpha_{i+1},\ \ t \ge t_{2i+1} \Leftrightarrow \\ \Leftrightarrow (m_i - F_{i+1}) e^{-m_i l_i T_1} \le \alpha_{i+1},\ i = \overline{1, 4}.$$

For ensuring the boundaries of the linear zones corresponding to the desired estimation accuracy (15), we consider (A.8) and the estimates of the observation errors $\varepsilon_i(t)$, $i = \overline{1, 4}$, for $t \ge t_{2i+1}$:

$$|\varepsilon_i| \le \frac{F_{i+1} + \alpha_{i+1}}{m_i l_i} + \alpha_i \le \delta_i,\ i = \overline{1, 4}. \quad (A.9)$$

For example, let $\alpha_i = \delta_i / 2$, $i = \overline{1, 4}$, and $\alpha_5 = \delta_5$. From the expressions (A.8) and (A.9) it follows that for $t \ge T$, both inequalities $|\varepsilon_i(t)| \le \delta_i$ and $|\varepsilon_{i+1}(t) - v_i(t)| \le \delta_{i+1}$, $i = \overline{1, 4}$, will hold for any $m_i \ge m_i^*$ (A.5) and $l_i \ge l_i^*$, where

$$l_i^* = \frac{1}{m_i} \max \left\{ \frac{2F_{i+1} + \delta_{i+1}}{\delta_i};\ \frac{1}{T_1} \ln \frac{2(m_i - F_{i+1})}{\delta_{i+1}} \right\},\ i = \overline{1, 3};$$

$$l_4^* = \frac{1}{m_4} \max \left\{ \frac{2(F_5 + \delta_5)}{\delta_4};\ \frac{1}{T_1} \ln \frac{m_4 - F_5}{\delta_5} \right\}.$$

The proof of this lemma is complete.

REFERENCES

1. Spong, M., Hutchinson, S., and Vidyasagar, M., *Robot Modeling and Control*, New York: Wiley, 2005.
2. Angeles, J., *Fundamentals of Robotic Mechanical Systems: Theory, Methods and Algorithms*, 3rd ed., New York: Springer, 2007.
3. Utkin, V.I., Guldner, J., and Shi, J., *Sliding Mode Control in Electromechanical Systems*, New York: CRC Press, 2009.
4. Krasnova, S.A., Utkin, V.A., and Utkin, A.V., Block Control Design for Mechanical Systems under Uncertainty, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2009, no. 6, pp. 41–54. (In Russian.)
5. Krasnova, S.A. and Kuznetsov, S.I., Uncontrollable Perturbations of Nonlinear Dynamic Systems: Estimation on Moving Modes, *Automation and Remote Control*, 2005, vol. 66, no. 10, pp. 1580–1593.
6. Krasnova, S.A., Estimating the Derivatives of External Perturbations Based on Virtual Dynamic Models, *Automation and Remote Control*, 2020, vol. 81, no. 5, pp. 897–910.
7. Krasnov, D.V. and Utkin, A.V., Synthesis of a Multifunctional Tracking System in Conditions of Uncertainty, *Automation and Remote Control*, 2019, vol. 79, no. 12, pp. 345–357.
8. Antipov, A.S., Krasnov, D.V., and Utkin, A.V., Decomposition Synthesis of the Control System of Electromechanical Objects in Conditions of Incomplete Information, *Mechanics of Solids*, 2019, vol. 54, no. 5, pp. 47–60.
9. Malikov, A.I., Synthesis of State Unknown Input Observers for Nonlinear Lipschitz Systems with Uncertain Disturbances, *Automation and Remote Control*, 2018, vol. 79, no. 3, pp. 406–424.
10. Nikiforov, V.O., Observers of External Deterministic Disturbances. P. II. Objects with Unknown Parameters, *Automation and Remote Control*, 2004, vol. 65, no. 11, pp. 1724–1732.
11. Golubev, A.E., Stabilization of Single-Link Manipulator with Incomplete State Measurement: Feedback by the Angular Coordinate of the Motor Shaft, *Science and Education. Scientific Edition of Bauman MSTU*, 2012, no. 11, pp. 395–412. (In Russian.)

12. Andrievsky, B.R. and Furtat, I.B., Disturbance Observers: Methods and Applications. P. II. Applications, *Automation and Remote Control*, 2020, vol. 81, no. 10, pp. 1775–1818.

13. Kokunko, Ju.G., Krasnov, D.V., and Utkin, A.V., Two Methods of Synthesis of State and Disturbances Observers for an Unmanned Aerial Vehicle, *Control Sciences*, 2020, no. 1, pp. 3–16. (In Russian.)

14. Utkin, V.A. and Utkin, A.V., Problem of Tracking in Linear Systems with Parametric Uncertainties under Unstable Zero Dynamics, *Automation and Remote Control*, 2014, vol. 75, no. 9, pp. 1577–1592.

15. Krasnova, S.A. and Utkin, A.V., Analysis and Synthesis of Minimum Phase Nonlinear SISO Systems under External Unmatched Perturbations, *Automation and Remote Control*, 2016, vol. 77, no. 9, pp. 1665–1675.

16. Busurin, V.I., Win, Y.N., and Zheglov, M.A., Effect of Linear Acceleration on the Characteristics of an Optoelectronic Ring Transducer of Angular Velocity and Its Compensation, *Optoelectronics, Instrumentation and Data Processing*, 2019, vol. 55, no. 3, pp. 309–316.

17. Spong, M., Modeling and Control of Elastic Joint Robots, *ASME Journal of Dynamic Systems, Measurement and Control*, 1987, vol. 109, pp. 310–319.

18. Kochetkov, S.A. and Utkin, V.A., DC Motor Control on the Base of Vortex Algoritm, *Control Sciences*, 2014, no. 5, pp. 20–27. (In Russian.)

This paper was recommended for publication by L.B. Rapoport, a member of the Editorial Board.

Received February 18, 2021, and revised April 30, 2021.
Accepted May 13, 2021.

Author information

Krasnov, Dmitry Valentinovich. Researcher, Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia, ✉ dim93kr@mail.ru

Antipov, Aleksei Semenovich. Researcher, Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia, ✉ scholess18@mail.ru

Cite this article

Krasnov, D.V. and Antipov, A.S., Designing a Double-Loop Observer to Control a Single-Link Manipulator under Uncertainty. *Control Sciences* **4,** 23–33 (2021). http://doi.org/10.25728/cs.2021.4.3

Original Russian Text © Krasnov, D.V. and Antipov, A.S., 2021, published in *Problemy Upravleniya*, 2021, no. 4, pp. 27–39.

DOI: http://doi.org/10.25728/cs.2021.4.4

MODELS OF MANAGING INDUSTRIAL ENTERPRISES UNDER AN UNSTABLE ENVIRONMENT AND TECHNOLOGICAL RE-EQUIPMENT

K.A. Korennaya[1], A.V. Hollay[2], and O.V. Loginovskiy[3]

[1]JSC Kuznetskie Ferrosplavy, Novokuznetsk, Russia,
[2,3]South-Ural State University, Chelyabinsk, Russia

[1]✉ kkris221@mail.ru, [2]✉ alexander@hollay.ru, [3]✉ loginovskiiov@susu.ru

Abstract. Under the ongoing process of economic globalization, Russian industrial enterprises are currently facing the need to compete in world markets characterized by an extreme degree of instability due to political, social, economic, and other factors. In such conditions, classical strategic management methods become ineffective, primarily due to no appropriate tools to consider and correct several factors increasingly affecting the performance of enterprises. For example, in view of the deteriorating environmental situation in the world (even interpreted as an environmental disaster), the governments of different countries and various public organizations are putting pressure on industrial enterprises to carry out technological re-equipment. This paper surveys the classical approaches to the strategic management of industrial enterprises and the assessment of their effectiveness using mathematical models. The disadvantages of these approaches are discussed. A strategic management model is proposed that considers a fluctuating demand for the products of industrial enterprises caused by periodic economic crises. In addition, a model is constructed for assessing the effectiveness of enterprises implementing technological transformations to minimize their environmental damage. The models can be used to improve strategic decision mechanisms for managing industrial enterprises.

Keywords: strategy, strategic management, enterprise management, industrial management, global instability, decision-making model, industry, cost management, technological re-equipment.

INTRODUCTION

In the era of globalization and the development of a single world market [1], the performance of industrial enterprises is determined not only by their internal technical and economic potential but also by external conditions (environment) [2]. In recent years, instability has sharply increased in the world due to political, social, and economic factors [3]. In 2020, the COVID-19 pandemic had a shocking effect on the economies of all countries [4, 5].

In such conditions, the complexity of managing various organizational and production structures has grown dramatically [6, 7]. This particularly applies to Russian industrial enterprises: along with global problems, domestic enterprises are experiencing unprecedented political pressure (the sanctions policy) from major world powers and international organizations [8, 9]. Sanctions aim at both Russian enterprises and their owners [10, 11].

In this regard, conducting a competitive struggle in international markets becomes vital for Russian manufacturers [12], which calls for increasing the effectiveness of their activities [13, 14]. At the same time, many Russian enterprises need technological re-equipment [15–17], which is significantly complicated by the current political situation around Russia [18].

It can be stated that under global instability, the quality of management will determine the viability of enterprises and their prospects for further development [19]. Note that in such conditions, the choice of approaches, models, and criteria for assessing the effectiveness of strategic and operational management of industrial enterprises becomes one of the main tasks of good management.

1. ANALYSIS OF APPROACHES TO STRATEGIC MANAGEMENT OF INDUSTRIAL ENTERPRISES AND ASSESSING THEIR EFFECTIVENESS

Nowadays, profit is the most common indicator for assessing the effectiveness of industrial enterprises. Profit is defined as the excess of the company's income over its expenses [20]. The disadvantages of using this indicator include short planning horizons (1–3 years), which creates significant risks for the enterprise. In particular, the time for re-equipping a large industrial enterprise can significantly exceed these periods. With the cyclical dynamics of the world economy [21, 22], this circumstance causes insolvency traps: an enterprise incurring losses during crisis periods cannot complete its technological re-equipment due to insufficient financial resources.

This approach is developed by expanding the number of indicators for assessing the effectiveness of the enterprise and forming a set of financial results to make decisions by the top management [23]. Such a set may include indicators of financial stability, solvency, profitability, resource efficiency, etc.

Management based on a set of financial indicators and short planning horizons leads to the problem of coordinating such indicators for stating the optimal enterprise management problem [24, 25]. Within this approach, the company's results for the current period are often compared with those for previous periods. After comparison, either a general conclusion is made (and a strategy for the company's behavior in the market is developed [26, 27]), or particular conclusions are drawn (and a set of measures for improving the company's performance is formed [28, 29]).

An attempt to solve the problems associated with using the enterprise profit as a target is an approach to increase the company's value. In 1938, J. B. Williams [30] proposed using the "internal" value of an investment asset (an enterprise) as such a measure, calculated by discounting future dividends (the total amount of funds received by enterprise stockholders). Later on, B. Graham [31] discriminated between the internal (fundamental) value and the external (asset price) one, which should converge to each other under market mechanisms. Thus, the company's management should strive to increase the fundamental value of the enterprise. As shown by W. Buffett, the internal value can be calculated as the discounted value of funds withdrawn from the business [32, 33].

Currently, the direction focused on the growth of the company's value is called Value-Based Management (VBM), and its main goal is to maximize the value of companies [34, 35]. The main disadvantage of this direction is no clear understanding of how to calculate the value of a company: different approaches involve particular methods and calculation formulas.

When assessing and analyzing the effectiveness of investment projects, the *discounted cash flow* (DCF) method is often used. Here, the objective function (target indicator) is the *net present value* (NPV) calculated by the formula

$$NPV = \sum_{t=1}^{n} \frac{CF_t}{(1+r)^t} - I_0,$$

where CF_t denotes the cash flow in period $t = 1,\ldots, n$; I_0 is the initial investment; r specifies the discount rate.

If $NPV > 0$, then the project's yield is higher than the invested capital. In other words, this project will increase the company's value by NPV. If $NPV < 0$, the project will reduce the company's value.

According to this approach, the growth of the company's value can be achieved either by increasing the cash flows received as a result of the project (CF_i), or by reducing the initial investment (I_0), or by decreasing the discount rate (e.g., by reducing risks).

There exist other methods for calculating the company's value, particularly the total yield of stockholders, the cash flow yield per unit investment, etc. [36, 37].

The methods mentioned above are based on calculating the company's future discounted cash flows and the weighted average capital value [38, 39].

A significant benefit of the approach aimed at increasing the company's value is a longer-term forecast [40] compared to the approach based on several financial indicators of the enterprise.

Note that despite its advantage, the former approach actually models the company's activities and is intended to identify the parameters with a favorable effect on the target level of the company's value. After forming and analyzing the company's value model, top managers focus on improving some parameters of the organization's performance: reducing costs, optimizing inventories, reducing the time of asset turnover, etc. [41].

Another direction of assessing the effectiveness of an enterprise is an approach focused on meeting the needs of stakeholders (Stakeholder Value, STV) [42, 43]. This concept considers the interests of the owners and many other subjects engaged, in one way or another, in projects implemented by the enterprise (managers, creditors, employees, trade unions, etc.). The problems of this approach lie in the choice of an appropriate criterion for assessing the degree of satisfaction of particular stakeholders (see the corresponding indicators in the table below), and the need to coordi-

nate (through appropriate mechanisms) the interests of different stakeholders for elaborating an enterprise management: in most cases, the interests of stakeholders can be multidirectional.

Parameters for assessing enterprise's effectiveness by different groups of stakeholders

Stakeholders	Parameters for assessing enterprise's effectiveness
Owners (stockholders, investors)	Yield per stock, stock price gains, total dividends
Managers	Enterprise's financial indicators determining the profit of managers: sales revenue, financial result, profitability level, asset turnover, market share, etc.
Creditors	Guaranteed repayment of loan obligations, the level of security of borrowed funds with assets
Staff	Remuneration, career growth
State authorities	Taxes and fees paid to the budget and extra-budgetary funds, employment of the population
Public organizations and unions (trade unions, mass media, political parties, etc.)	Compliance with environmental safety requirements, maintaining political stability and loyalty, etc.

As an enterprise's effectiveness criterion, such approaches adopt either an integral (aggregated) indicator for the disparate interests of different stakeholders [44] or an indicator reflecting an agreed assessment of their interests [20]. For example, the stakeholders are ranked based on the importance of satisfying their interests.

In addition, there are mixed approaches to assessing the effectiveness of manufacturing companies. For example, M. Jensen [45] suggested a hybrid model considering the interests of different stakeholders: the idea is to maximize the company's value taking into account the interests of other stakeholders.

This approach suffers from the multicriteria objective function, which is difficult or even impossible to quantify. In the paper [46], the interests of different stakeholders were considered through the iterative coordination of their opinions.

Another drawback of this concept is the need to consider the interests of participants not investing in the company's development: the rights of investors and owners are discriminated against, which may reduce investment.

Thus, effective management of industrial enterprises should be based on the models of future cash flows generated by the management of these enterprises. Moreover, the mathematical models of industrial enterprise management should include various factors with the greatest effect on the enterprise's activities in the corresponding periods.

2. A MODEL OF MANAGING INDUSTRIAL ENTERPRISES IN A STABLE ENVIRONMENT

Under small fluctuations of demand and supply in world markets, the objective function (Φ_1) describing the efficiency of an industrial enterprise [20] can be the profit on sales (Π^T) in period T:

$$\Phi_1 = \Pi^T \to \max.$$

The profit on sales in period T is calculated as

$$\Pi^T = \sum_f I_f^T C_f^T - \sum_f \left(I_f^T \left(\sum_n S_{fn}^{ET} + \sum_n \sum_m S_{fnm}^{MT} + \sum_k \sum_n Z_{fkn}^T \right) \right) - S^{OT}, \quad (1)$$

where C_f^T denotes the unit price of product f manufactured in period T; I_f^T is the total output of product f in period T; S_{fn}^{ET} specifies the electrical energy costs per unit of product f manufactured on machine n in period T; S_{fnm}^{MT} indicates the costs of material m per unit of product f manufactured on machine n in period T; Z_{fkn}^T is the total wages of production workers in gang k per unit of product f manufactured on machine n in period T; finally, S^{OT} gives the total fixed costs of the industrial enterprise in period T.

Thus, in a stable environment, an enterprise should maximize profit primarily by increasing output and, in addition, reducing variable and fixed production costs (management costs, security costs, payments for environmental damage, etc.).

3. A MODEL OF MANAGING INDUSTRIAL ENTERPRISES UNDER FLUCTUATING DEMAND FOR PRODUCTS

We emphasize that the modern capitalist system is characterized by a cyclical demand for industrial products due to various imbalances and contradictions. They are determined by the characteristic features of capitalist production [21].

Under an increased demand for its products, the company seeks to raise outputs, adapting to market demands. The enterprise's management makes this operational decision based on the current assessment

of the market conditions and production capabilities. Hence, the expression for calculating the enterprise's profit can be written as

$$\Pi^T = \sum_f \left(\left(I^T_{f\text{cont}} + I^T_{f\text{add}} \right) C^T_f \right) - \sum_f \left(I^T_{f\text{cont}} + I^T_{f\text{add}} \right) \times$$
$$\times \left(\sum_n S^{ET}_{fn} + \sum_n \sum_m S^{MT}_{fnm} + \sum_k \sum_n Z^T_{fkn} \right) - S^{OT},$$

where $I^T_{f\text{cont}}$ denotes the total output of product f manufactured by the enterprise according to the contractual obligations in period T; $I^T_{f\text{add}}$ is the additional output of product f in period T.

Since the fixed costs S^{OT} remain unchanged in favorable periods, an increase in production will lead to profit growth if the profitability of product f is positive.

Profit should be maximized subject to several constraints, in particular: the output of product f must be not smaller than the volumes stipulated by the contracts; the outputs cannot exceed the technological capabilities of the enterprise. There may exist other constraints for a specific industrial enterprise.

Under a dropped demand for its products due to the world economic crisis, the enterprise seeks to minimize production costs instead of maximizing profits [47, 48]. In this case, the objective function can be written as

$$\Phi_2 = \min F\left(S^{ET}, S^{MT}, Z^T, P^T_d \right),$$

where S^{ET} denotes the electrical energy costs in period T; S^{MT} is the material costs in period T; Z^T specifies the remuneration costs in period T; P^T_d indicates the unexpected losses in period T due to the economic crisis effect.

In other words, the objective function Φ_2 takes the form

$$\Phi_2 = \sum_f \left(I^T_f \sum_n \left(S^{ET}_{fn} + S^{MT}_{fnm} + Z^T_{fkn} \right) \right) + P^T_d \to \min \cdot$$

The enterprise's activities in period T are assessed by analyzing the total costs in this period:

$$\Phi^{\circ}_2 = \sum_T \left(S^{ET} + S^{MT} + Z^T + P^T_d \right) \cdot$$

During this period, a possible management decision is decreasing the output to reduce the total costs. Note that, first of all, the outputs of products with a negative marginal profit should be decreased. At the same time, in some industries (e.g., metallurgy), it is impossible to lower the output below a certain threshold due to technological restrictions: the production process cannot be resumed after stop. All these features should be considered when constructing mathematical models of particular industrial enterprises.

Thus, under a significantly fluctuating demand for products during economic crises, the objective function of an industrial enterprise is to reduce the total production and non-production costs while fulfilling all contractual obligations.

In this case, the company's stockholders need timely forecasts of possible crisis phenomena to make economically sound managerial decisions, allowing the company to adapt to abrupt exogenous changes.

4. A MODEL OF MANAGING INDUSTRIAL ENTERPRISES WITH TECHNOLOGICAL RE-EQUIPMENT AND EMISSION REDUCTION

Note that the environmental damage caused by production is an important factor affecting the demand for industrial products. If an enterprise uses technologies with a significant impact on the environment, the society responds by mechanisms forming negative public opinion about the consumption of its products manufactured using "dirty" technologies.

As an illustrative example, consider the concept of carbon footprint. Carbon footprint refers to the greenhouse gas emissions from production associated with fuel combustion, separate industrial processes, agriculture, etc. For example, the manufacture of stamped metal parts for cars emits a small volume of carbon dioxide, in contrast to metal production for such parts. Also, an industrial enterprise consumes a large amount of electricity, which can be generated at various types of power plants (CHP, NPP, HPP, etc.), leaving behind a greater or lesser carbon footprint.

The ISO 14061-14064 standard is the most commonly used method for determining the carbon footprint in the world. In Russia, separate guidelines exist for quantifying the volume of greenhouse gas emissions; see orders no. 300 of June 30, 2015, and no. 330 of June 29, 2017, of the Ministry of Natural Resources and Environment of the Russian Federation.

As a result, the carbon footprint of products includes the total volume of emissions of the enterprise and its main suppliers. An enterprise consuming electricity from fuel power plants to manufacture its products will generate a larger carbon footprint than the one consuming electricity from cleaner power plants with comparable technologies.

When choosing suppliers, many large enterprises are guided by the volume of their carbon footprint. For example, Volkswagen has introduced an environmental rating for suppliers; Hewlett Packard Enterprise sets requirements for suppliers to reduce greenhouse gas emissions; Walmart is decreasing the number of suppliers with a large carbon footprint [49].

Thus, industrial enterprises will face an acute need to carry out technological re-equipment in the near future due to the transition to environmentally friendly technologies. (Otherwise, they will suffer from reduced demand for their products.)

The models considered above, particularly formula (1), overestimate the forecasted profit by neglecting the drop in demand due to the enterprise's technological lagging. To compensate for this overestimation, we introduce a correction coefficient into formula (1):

$$\Pi_g^T = \sum_f I_f^T C_f^T g_f^T \sum_f \left(I_f^T g_f^T \left(\sum_n S_{fn}^{ET} + \right. \right. \\ \left. \left. + \sum_n \sum_m S_{fnm}^{MT} + \sum_k \sum_n Z_{fkn}^T \right) \right) - S^{OT}, \quad (2)$$

where Π_g^T is the enterprise's profit on sales in period T considering the drop in demand due to using "dirty" technologies; g_f^T is a coefficient showing how many times the demand for product f will drop in period T due to using "dirty" technologies by the enterprise.

It should be understood that $\Pi_g^T < \Pi^T$. With the constant total costs S^{OT}, a decrease in demand can make the enterprise unprofitable in period T, whereas the classical model will predict profit.

Next, we study decision-making on the enterprise's technological re-equipment, transforming the expression (1) under several assumptions:

– Re-equipment will require additional investment, which reduces free retained profits.

– The demand for products will not change since the company eliminates environmentally "dirty" technologies ($g_f^T = 1$).

– If the adoption of new technologies changes power consumption, material consumption, and the number of required personnel, this should be considered by introducing appropriate correction coefficients.

Then the mathematical model of the company's profit with technological re-equipment can be written as

$$\Pi_{tech}^T = \sum_f I_f^T C_f^T - \sum_f \left(I_f^T \left(\sum_n S_{fn}^{ET} e_n^{ET} + \right. \right. \\ \left. \left. + \sum_n \sum_m S_{fnm}^{MT} r_{nm}^{MT} + \sum_k \sum_n Z_{fkn}^T p_{kn}^T \right) \right) - S^{OT} - \sum_f F_f^T, \quad (3)$$

where F_f^T is the re-equipment investment for product f in period T; e_n^{ET} denotes the coefficient of electrical energy consumption variation per unit of product f manufactured on machine n in period T; r_{nm}^{MT} denotes the coefficient of consumption variation for material m per unit of product f manufactured on machine n in period T; p_{kn}^T denotes the coefficient of variation of the total wages of production workers in gang k per unit of product f manufactured on machine n in period T.

The correction coefficients g_f^T, e_n^{ET}, r_{nm}^{MT}, and p_{kn}^T are determined using expertise depending on the macroeconomic forecast.

Recall that formulas (1)–(3) represent the profit evaluated for period T, often one year. However, technological re-equipment and its economic effect should be assessed over a longer period. Therefore, when making decisions, it is necessary to consider the total amount of profit (cash flows) for the compared period, i.e.,

$$\sum_T \Pi_g^T = \sum_T \left[\sum_f I_f^T C_f^T g_f^T - \sum_f \left(I_f^T g_f^T \times \right. \right. \\ \left. \left. \times \left(\sum_n S_{fn}^{ET} + \sum_n \sum_m S_{fnm}^{MT} + \sum_k \sum_n Z_{fkn}^T \right) \right) - S^{OT} \right],$$

$$\sum_T \Pi_{tech}^T = \sum_T \left[\sum_f I_f^T C_f^T - \sum_f \left(I_f^T \left(\sum_n S_{fn}^{ET} e_n^{ET} + \right. \right. \right. \\ \left. \left. \left. + \sum_n \sum_m S_{fnm}^{MT} r_{nm}^{MT} + \sum_k \sum_n Z_{fkn}^T p_{kn}^T \right) \right) - S^{OT} - \sum_f F_f^T \right].$$

Thus, we write two problem statements of the maximization problem:

- without technological re-equipment (the enterprise's cash flow in period T is estimated without technological re-equipment and under a dropped demand due to using "dirty" technologies):

$$\Phi_3 = \sum_T \Pi_g^T \to \max;$$

- with technological re-equipment (the enterprise's cash flow in period T is estimated after elimination of "dirty" technologies and under the same demand for products ($g_f^T = 1$, and the current cost coefficients (e_n^{ET}, r_{nm}^{MT}, and p_{kn}^T) may even decrease):

$$\Phi_4 = \sum_T \Pi_{tech}^T \to \max.$$

The decision about technological re-equipment should be made if $\Phi_4 > \Phi_5$:

$$\Phi_5 = \max\{\Phi_3; \Phi_4\} = \max\left\{ \sum_T \Pi_g^T; \sum_T \Pi_{tech}^T \right\}.$$

In other words, for the period under consideration, the total forecasted cash flow with technological re-equipment should exceed the one in the case of preserving the existing production technologies.

CONCLUSIONS

When implementing strategic management, the enterprise's owners and managers must first choose an appropriate effectiveness criterion for assessing managerial decisions. This paper has presented ways and methods for constructing mathematical models of assessing the effectiveness of industrial enterprises in an unstable environment. In particular, models of assessing the effectiveness of activities with fluctuating demand for products and the need for technological reequipment have been constructed. These factors have the greatest effect on the performance of modern enterprises. The general logic of modeling proposed above can be used to consider other significant factors that may affect industrial enterprises in future.

REFERENCES

1. Korennaya, K.A., Loginovskiy, O.V., and Maksimov, A.A., *Upravlenie promyshlennymi predpriyatiyami v usloviyakh global'noi nestabil'nosti* (Management of Industrial Enterprises under Global Instability), Chelyabinsk: South-Ural State University, 2013. (In Russian.)
2. Korennaya, K.A., Loginovskiy, O.V., Maksimov, A.A., and Zimin, A.V., *Global Economic Instability and Management of Industrial Organisations*, Shestakov, A.L., Ed., Kostanay: Kostanay State University, 2014.
3. Loginovskiy, O.V., Maksimov, A.A., and Khaldin, K.S., Management of Material Resources of an Industrial Enterprise in Modern Conditions, *Dinamika Slozhnykh Sistem*, 2016, no. 2, pp. 33–38. (In Russian.)
4. Ibragimova, E.S., Vatsaev, B.Sh., and Shamaev, R.A., Problems of Economic Development of the Russian Federation and Socio-Economic Consequences of COVID-19, *Moskovskii Ekonomicheskii Zhurnal*, 2021, no. 1, pp. 111–116. (In Russian.)
5. Bokonbaev, K.Dzh., Some Considerations about the COVID-19 Pandemic and Its Consequences, *Vestnik KRSU*, 2020, vol. 20, no. 8, pp. 94–97. (In Russian.)
6. Novikov, D.A., *Control Methodology*, New York: Nova Science, 2013.
7. Novikov, D.A., *Theory of Control in Organizations*, New York: Nova Science, 2013.
8. Gordienko, D.V., American Sanctions Policy towards Russia, *Economics and Management: Problems, Solutions*, 2019, vol. 5, no. 1, pp. 16–31. (In Russian.)
9. Tishchenko, A. and Dyakina, D., The Impact of Sanctions Policy on the Competitiveness of Russia, *Central Problems of Innovation Economics*, 2015, no. 12, pp. 23–29. (In Russian.)
10. Glaz'ev, S.Yu. and Arkhipova, V.V., Sanctions and Other Crisis Factors Impact Assessment on the Russian Economy's State, *Russian Economic Journal*, 2018, no. 1, pp. 3–29. (In Russian.)
11. Kharlanov, A.S. and Chereshneva, K.K., The Mechanism of Sanctions Pressure as a Means of Competition, *Journal of Economy and Entrepreneurship*, 2019, no. 6 (107), pp. 834–839. (In Russian.)
12. Okunev, V.I., Strengthening Global Competition as a Long-Term Systemic Challenge, *The World Economics*, 2019, no. 2, pp. 68–76. (In Russian.)
13. Alekhina, O.F., Voronov, N.A., and Udalov, F.E., *Operativnoe i strategicheskoe upravlenie proizvodstvom na promyshlennykh predpriyatiyakh* (Operational and Strategic Production Management at Industrial Enterprises), Nizhny Novgorod: Lobachevsky State University, 2013. (In Russian.)
14. Korennaya, K.A., Loginovskiy, O.V., and Maksimov, A.A., The Strategy for the Effective Management of Large Industrial Enterprises, *Bull. South-Ural State Univ. Ser. Comp. Techn., Autom. Control, Radio Electr.*, 2016, vol. 16, no. 3, pp. 102–109. (In Russian.)
15. Glaz'ev, S.Yu., *Razvitie Rossiiskoi ekonomiki v usloviyakh global'nykh tekhnologicheskikh sdvigov: nauchnyi doklad* (Development of the Russian Economy in the Context of Global Technological Shifts: Scientific Report), Moscow: National Institute of Development, 2007. (In Russian.)
16. Sukhorukov, A.V., Strategic Planning of Innovative Development of an Industrial Enterprise, *Rossiiskoe Predprinimatel'stvo*, 2014, vol. 15, no. 8, pp. 64–70. (In Russian.)
17. Hollay, A.V. and Loginovskiy, O.V., Managerial Decision-Making Model Taking into Account Technological Development of the Enterprise, *Bull. South-Ural State Univ. Ser. Comp. Techn., Autom. Control, Radio Electr.*, 2017, no. 4, pp. 142–145. (In Russian.)
18. Aturin, V.V., Anti-Russian Economic Sanctions and Problems of Import Substitution in the Context of Modern International Competition, *The Eurasian Scientific Journal*, 2019, no. 2. (In Russian.)
19. Loginovskiy, O.V. and Korennaya, K.A., Increasing the Efficiency of the Enterprise on the Basis of Modern Management Technologies and Information-Analytical Systems, *Trudy Vserossiiskogo foruma "Informatsionnoe obshchestvo – 2015: vyzovy i zadachi"* (Proc. The All-Russian Forum "Information Society – 2015: Challenges and Tasks"), Chelyabinsk: South-Ural State University, 2015, pp. 4–38. (In Russian.)
20. Loginovskii, O.V., Maksimov, A.A., Burkov, V.N., Burkova, I.V., Gel'rud, Ya.D., Korennaya, K.A., and Shestakov, A.L., *Upravlenie promyshlennymi predpriyatiyami: strategii, mekhanizmy, sistemy* (Industrial Enterprise Management: Strategies, Mechanisms, Systems), Moscow: INFRA-M, 2018. (In Russian.)
21. Kondrat'ev, N.D., Yakovets, Yu.V., and Abalkin, L.I., *Bol'shie tsikly kon''yunktury i teoriya predvideniya: izbrannye trudy* (Big Cycles of the Conjuncture and the Theory of Foresight: Selected Works), Moscow: Ekonomika, 2002. (In Russian.)
22. Glaz'ev, S.Yu., Modern Theory of Long Waves of Economic Development, *Economics of Contemporary Russia*, 2012, no. 2 (57), pp. 27–42. (In Russian.)
23. Loginovskiy, O.V., Dranko, O.I., and Hollay, A.V., Mathematical Models for Assessment the Activities of Industrial Enterprises under Conditions of Instability, *Bull. South-Ural State Univ. Ser. Comp. Techn., Autom. Control, Radio Electr.*, 2018, vol. 18, no. 4, pp. 88–102. (In Russian.)
24. Burkov, V.N., Goubko, M.V., Korgin, N.A., and Novikov, D.A., *Introduction to Theory of Control in Organizations*, CRC Press, 2019.
25. Goubko, M., Burkov, V., Kondrat'ev, V., Korgin, N., and Novikov, D., *Mechanism Design and Management: Mathemat-*

ical Methods for Smart Organizations, New York: Nova Science, 2013.
26. Knysh, M.I., Puchkov, V.V., and Tyutikov, Yu.P., *Strategicheskoe upravlenie korporatsiyami* (Strategic Management of Corporations), St. Petersburg: Kul't-inform-press, 2002. (In Russian.)
27. Koch, R., *Strategy: How to Create, Pursue and Deliver a Winning Strategy*, 4th ed., Financial Times/Prentice Hall, 2012.
28. Rice-Johnston, W., *Tactical Management: A Management Model for Challenging Times*, 1st ed., Cengage Learning Emea, 1999.
29. Thomas, R., *Quantitative Methods for Business Studies*, 1st ed., Prentice Hall, 1997.
30. Williams, J., *The Theory of Investment Value*, Boston: Harvard Univ. Press, 1938.
31. Graham, B., *The Intelligent Investor: The Definitive Book on Value Investing*, Harper Business, 2006.
32. *The Essays of Warren Buffett: Lessons for Corporate America*, Selected, Arranged and Introduced by L.A. Cunningham, 4th ed., The Cunningham Group & Carolina Academic Press, 2015.
33. Purlik, V.M., Corporate Management Focused on Business Value Growth, *Public Administration*, 2018, no. 69, pp. 127–154. (In Russian.)
34. Evans, F.C. and Mellen, C.M., *Valuation for M&A: Building and Measuring Private Company Value*, New Jersey: Wiley, 2018.
35. Scales, D., *The M&A Solution: A Values-Based Approach to Integrate Companies*, Charleston: Forbes Books, 2021.
36. Kudina, M.V., A Cost Approach to Company Management, *Vestn. Mosk. Univ. Ser. Upravl. (Gosud. Obshch.)*, 2007, no. 3, pp. 61–81. (In Russian.)
37. Koller, T., Goedhart, M., and Wessels, D., *Valuation: Measuring and Managing the Value of Companies*, New Jersey: Wiley, 2020.
38. Gaughan, P.A., *Mergers, Acquisitions, and Corporate Restructurings*, 7th ed., Wiley, 2017.
39. Barulina, E.V. and Barulin, S.V., *Upravlenie stoimost'yu kompanii: finansovyi kontrolling, menedzhment, informatsionno-servisnoe obespechenie* (Company Value Management: Financial Controlling, Management, Information and Service Support), Moscow: Rusains, 2016. (In Russian.)
40. Eccles, R.G., Herz, R.H., Mary Keegan, E., and Phillips, M.H.D., *The ValueReporting Revolution: Moving Beyond the Earnings Game*, New York: Wiley, 2001.
41. Dranko, O., The Aggregate Model of Business Valuation by Three Methods, *Proceedings 13th International Conference "Management of Large-Scale System Development" (MLSD'2020)*, Moscow: Trapeznikov Institute of Control Sciences and IEEE, 2020, pp. 1–4. DOI: 10.1109/MLSD49919.2020.9247842 (In Russian.)
42. Ivashkovskaya, I.V., Stakeholder Approach to Value-Based Management, *Journal of Corporate Finance Research*, 2012, vol. 6, no. 1(21), pp. 14–23. (In Russian.)
43. Mabelo, P.B., *How to Manage Project Stakeholders: Effective Strategies for Successful Large Infrastructure Projects*, London: Routledge, 2020. DOI 10.4324/9781003023791.
44. Ivashkovskaya, I.V., *Finansovye izmereniya korporativnykh strategii. Steikkholderskii podkhod* (Financial Dimensions of Corporate Strategies. A Stakeholder Approach), Moscow: INFRA-M, 2016. (In Russian.)
45. Jensen, M., Value Maximization, Stakeholders Theory, and the Corporate Objective Function, *Journal of Applied Corporate Finance*, 2001, vol. 14, no. 3, pp. 8–21.
46. Gel'rud, Ya.D. and Loginovskiy, O.V., *Upravlenie proektami: metody, modeli, sistemy* (Project Management: Methods, Models, Systems), Shestakov, A.L., Ed., Chelyabinsk: South-Ural State University, 2015. (In Russian.)
47. Womack, J.P. and Jones, D.T., *Lean Thinking: Banish Waste and Create Wealth in Your Corporation*, 2nd ed., New York: Free Press, Simon & Schuster, 2003.
48. Hollay, A.V., Agile Project Management as a Way to Combat Losses in Production, *Trudy mezhdunarodnoi nauchno-prakticheskoi konferentsii "Novye tekhnologii v promyshlennosti, nauke i obrazovanii"* (Proc. Int. Sci.-Pract. Conf. "New technologies in Industry, Science, and Education"), Orenburg, October 28, 2017, Sterlitamak: Agency of International Studies, 2017, pp. 35–39. (In Russian.)
49. URL:https://icss.ru/otrasli-i-ryinki/energeticheskij sektor/uglerodnyy-sled-rossiyskoy-energetiki

This paper was recommended for publication by V.N.Burkov, a member of the Editorial Board.

Received May, 18, 2021,
and revised June, 16, 2021.
Accepted June, 22, 2021

Author information

Korennaya, Kristina Aleksandrovna. Cand. Sci. (Eng.), JSC Kuznetskie Ferrosplavy, Novokuznetsk, Russia,
✉ kkris221@mail.ru

Hollay, Aleksandr Vladimirovich. Dr. Sci. (Eng.), South-Ural State University, Chelyabinsk, Russia,
✉ alexander@hollay.ru

Loginovskiy, Oleg Vital'evich. Dr. Sci. (Eng), South-Ural State University, Chelyabinsk, Russia,
✉ loginovskiiov@susu.ru

Cite this article
Korennaya, K.A., Hollay, A.V., Loginovskiy, O.V. Models of Managing Industrial Enterprises under an Unstable Environment and Technological Re-equipment, *Control Sciences* **4,** 34–40 (2021). http://doi.org/10.25728/cs.2021.4.4

Original Russian Text © Korennaya, K.A., Hollay, A.V., Loginovskiy, O.V., 2021, published in *Problemy Upravleniya*, 2021, no. 4, pp. 40–49.

DOI: http://doi.org/10.25728/cs.2021.4.5

ESTIMATING TIME CHARACTERISTICS OF CONTROL SYSTEMS WITH CYCLIC OPERATION: A NETWORK CALCULUS APPROACH[1]

V.G. Promyslov[2] and K.V. Semenkov[3]

Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia

[2]✉ v1925@mail.ru, [3]✉ semenkovk@mail.ru

Abstract. The practical validation of time characteristics of digital control systems is considered. The delay in information processing and transmission often has a probability distribution differing from the Gaussian one. Therefore, the confidence intervals calculated under the Gaussian distribution assumption will be incorrect for such systems. The idea is to estimate the time characteristics of a control system using non-statistical time parameter estimation methods. As one of such methods, Network Calculus is considered. The practical implementation of Network Calculus to estimate the parameters of control systems, particularly its features, is discussed. One of the main features is imposing special restrictions on data flows and system performance, determined by flow envelopes and maximum (minimum) service curves. Generally, these characteristics are unknown in advance. Mathematical methods are proposed to estimate these characteristics under known input and output data flows in the system. As shown below, the calculation of characteristics is significantly simplified for systems with cyclic data processing algorithms, and the data transfer rate over the network is much higher than that on the computing components of the system. Simulations are carried out, and the system's time parameters estimated by Network Calculus are compared with the results of classical statistical estimation methods. As an illustrative example, the time parameters of one component of a real nuclear power plant instrumentation and control system are estimated using Network Calculus.

Keywords: system performance, time characteristics, digital control systems, Network Calculus, non-statistical estimation methods.

INTRODUCTION

In most cases, modern control systems (CSs) for industrial plants are implemented as a computing environment distributed by functions and means. Its components are a set of hardware and software means for acquisition, accumulation, asynchronous processing, representation, and transmission of information. CS components can be distributed both spatially and functionally.

Dynamic validation methods of system parameters have been developed to confirm the CS operability. They vary depending on the industry: for example, the IAEA methodology is used for CSs of nuclear power plants (NPPs) [1]. Network simulators based on discrete mathematics, such as OMNeT ++ [1–3] and OPNET [4, 5], are used to estimate the dynamic characteristics of CSs. However, statistical methods are necessarily applied to confirm the time characteristics of a real system. In the latter case, the samples of measurements of the system's parameters are analyzed under the common assumption that the distribution law of the measured characteristics is close to Gaussian [1]. In most cases, this assumption is true for physical signals. As shown below, it can be false for the parameters describing the digital CS itself, e.g., signal transmission and processing times.

Network Calculus [6], a non-statistical analysis method for deterministic systems, is an alternative approach to estimating the characteristics of data flows between the components of computer networks. This method is based on the min-plus algebra and is attractive: in many cases, it allows considering linear systems that are nonlinear in the conventional algebra.

[1]This work was supported in part by the Russian Foundation for Basic Research, project no. 19-29-06044 (Sections 2 and 3).

The method involves no assumptions about the type of distribution for the measured process.

A feature of Network Calculus is using specific functions – the envelopes of the input and output data flows and service curves – o calculate the system characteristics, which primarily include the data transmission delay and buffering parameters.

Network Calculus was developed for analyzing flow systems without information losses during processing (e.g., for calculating the throughput of a network segment or determining the parameters of video streaming over Ethernet networks). Generally speaking, control systems do not belong to this class of systems due to the following characteristics:

- parallel processing of several tasks on one computing resource,
- a significant change in the volume of information at the component's input and output (the output flow can be either greater or smaller than the input one, e.g., when compressing information),
- heterogeneous information in CSs, unlike information transmission systems (here, information heterogeneity means that each element (bit) has a specific value and can be processed according to a particular algorithm).

No doubt, these features were considered within Network Calculus. The authors [7] extended the method to systems with cyclic dependencies between the input and output flows of components. The papers [8, 9] presented Network Calculus approaches for systems with a significant change in the input-output flow ratio. The publications [10, 11] considered various methods for describing joint processing disciplines of several tasks on one computing resource.

The approaches mentioned above have common drawbacks. First of all, their application requires accurate knowledge of the internal features of system operation: bound to them, the approaches become sensitive to any change in the operating modes of the system. In addition, when used for complex systems, the approaches lose the "transparency" of results and their simple correlation with other characteristics (the input data rate, data unevenness, and the computing power of the component).

In view of these drawbacks, when constructing a system model below, we will try preserving the generality and transparency of the results (on the one hand) and reflecting the unevenness of input and output data flows and the dependence of the data processing algorithm on the information contained in them (on the other hand).

Also, we will validate Network Calculus by comparing the delays yielded by statistical estimation methods and the former method. This comparison is of particular interest for correlating the results of Network Calculus with those of statistical methods, which is an underinvestigated problem.

We will solve the problem on model examples of control systems and a real NPP instrumentation and control system [11]. The CS under study is of a rather general type, and the problems considered below are common for the developers of industrial CSs. Therefore, this experience may be of interest to other researchers and engineers of industrial CSs.

1. THE STRUCTURE OF A TYPICAL PROCESS CONTROL SYSTEM

1.1. Typical interaction of control system components

This paper considers a typical CS for an industrial plant, further referred to as the CS. A similar CS structure arises in various applications for real plants; see [11, 13].

The typical structure of the CS is shown in Fig. 1. There are three levels:

– programmable logic controllers (PLC) and gateways (level *G* or level 1),

– the servers of primary processing and data storage (level *S* or level 2), and

– the components providing the human-machine interface (level *Z* or level 3).

The CS architecture under consideration has the following properties:

– One or more components of levels 1 and 3 can be connected to the server.

– Each communication channel (C1–C4) between different-level components can be redundant (redundancy is not shown in the diagram).

– Components *G, S,* and *Z* have cyclic information processing algorithms.

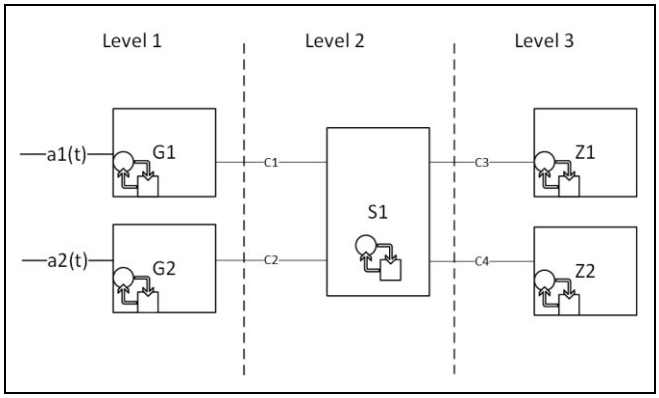


Fig. 1. The structural diagram of control system.

1.2. Data processing features

According to the operating conditions of the CS (see Fig. 1), different-level elements of the system use cyclic processing algorithms for the data transmitted from the gateway (*G*) at the level of connection to the server (*S*) and from the server to the workstation (*Z*). The data are the signals representing the plant's state and the control system itself.

Definition 1. A system element implements a cyclic data processing algorithm if the algorithm has the following properties:

– The element's initial state is waiting for the data arrival.

– The sequentially incoming data packets are processed in a deterministic uniform way, after which the system returns to the initial state.

For a cyclic algorithm, the total processing time of a data packet can be written as the sum of two values:

$$D_C = T_E + T_S ,$$

where T_E denotes the network delay, and T_S is the processing time on the CS element.

2. NETWORK CALCULUS FOR ESTIMATING TIME CHARACTERISTICS OF CONTROL SYSTEMS

2.1. Foundations of Network Calculus

Network Calculus [6] is based on rather new methods of applied mathematics introduced by Cruz [14, 15]. They involve the min-plus algebra; see the monograph [16]. The main application area of Network Calculus is the studies of queuing systems.

Let us briefly describe the method, following the book [6].

Definition 2. A flow (also called a cumulative flow) is a nondecreasing function of time such that

$$\begin{cases} A(t) \le A(s), \forall t < s, \\ A(t) \in \mathbb{R}_+ \cup \{+\infty\}, \\ t \in \mathbb{R}. \end{cases}$$

A flow is said to be causal if $A(t) = 0,\ \forall t \le 0$. The set of all causal functions is often denoted by *F*. For such functions, we introduce the operations of convolution and deconvolution.

Definition 1. Let A and β be causal flows. Their min-plus convolution, denoted by $A^* = A \otimes \beta$, has the form

$$A^*(t) = \inf_{0 \le s \le t} \{\beta(t-s) + A(s)\}. \qquad (1)$$

Whenever no confusion occurs, we will omit the variable t in the expressions below. Obviously, $A^*(t) = 0,\ \forall t < 0$, and A^* is nonnegative since both A and β are nonnegative causal functions.

Definition 4. Consider functions A and β, where β is causal. Their mini-plus deconvolution, denoted by $H = A \oslash \beta$, has the form

$$H = \sup_{u \ge 0} \{A(t+u) - \beta(u)\}. \qquad (2)$$

Note that the deconvolution of flows A and β, where β is causal, is a flow as well.

Definition 5. Let A and β be causal functions. Their max-plus convolution, denoted by $A^* = A \overline{\otimes} \beta$, has the form

$$A^*(t) = \sup_{0 \le s \le t} \{\beta(t-s) + A(s)\}. \qquad (3)$$

Definition 6. Let A and β be flows, where β is causal. Their maxi-plus deconvolution, denoted by $H = A \overline{\oslash} \beta$, has the form

$$H = \inf_{u \ge 0} \{A(t+u) - \beta(u)\}. \qquad (4)$$

Definition 7. A function β is the (minimum) service curve of a network element (or a system) with an input flow A if β is a causal flow and the output flow A^* of the element (system) satisfies the relation

$$A^* \ge A \otimes \beta. \qquad (5)$$

Definition 8. A function γ is the (maximum) service curve of a network element (or a system) with an input flow A if γ is a causal flow and the output flow A^* of the element (system) satisfies the relation

$$A^* \le A \otimes \gamma. \qquad (6)$$

Definition 9. A function a is called an envelope of a flow A if $A \le A \otimes a$ or, equivalently,

$$a \ge A \oslash A. \qquad (7)$$

Incoming and outgoing flows are determined by the total volume of data observed at the input and output over a certain period. Therefore, the data pass through the system in a time defined as the horizontal deviation between these functions, $d(t)$.

Definition 10 (maximum delay in system). For linear systems with an input flow A, an output flow A^*, $A(t) \ge A^*(t)$, the maximum delay D_{max} is the maximum horizontal distance between the input and output flows:

$$D_{\max} = h(A, A^*) = \sup_{t \ge 0} \{\inf \{d \ge 0 : A(t) \le A^*(t+d)\}\}.$$

A fundamental result of Network Calculus is the possibility of determining delays using flow envelopes and service curves instead of cumulative flows:

$$D_{\max} = h(a, \beta). \qquad (8)$$

It was proved in the book [6].

3. APPROACHES TO ESTIMATING CHARACTERISTIC CURVES OF NETWORK CALCULUS

Consider the problem of determining the flow envelope, the system's maximum and minimum service curves, and their linear approximations based on experimentally measured flows.

3.1. Calculating flow envelope based on experimental data

Equation (7) gives a direct way to calculate the envelope of a cumulative flow *A*. For facilitating calculations, it is convenient to operate the piecewise-linear approximation of the envelope, reduced in some cases to the affine function $y = kx + b$. The piecewise-linear representation allows adopting efficient computational algorithms for data processing. Operating the affine function, we can quickly ("on the fly") analyze the system and assess its behavior in quantitative terms.

Piecewise-linear approximation is traditionally used in the analysis of complex systems. Due to simplicity, this model is indispensable for Network Calculus, where both specially developed algorithms [10] and mathematical methods of optimal control and system identification are applied; for example, see [17] or [18].

The approximation of flow envelopes by the affine curve within Network Calculus was considered in [19]. The methods for calculating a one-component linear flow envelope suggested therein were based on support vector algorithms [20].

3.2. Calculating maximum and minimum service curves based on experimental data

Formula (7) allows directly calculating the flow envelope: the problem reduces to searching for effective analytical and computer methods of linear approximation. Determining the parameters of the service curve is much more difficult.

Theoretically, an exact estimate of the service curve can be obtained from formulas (1) and (2) using a specially selected test flow and the fact that the zero element δ_0 of the convolution functions is absorbed by the operator $\otimes$; see the book [6], p. 111. However, such an experiment is impracticable: it requires generating an infinitely large flow, which exceeds the capabilities of any real system.

The second approach is based on the property of the min-plus algebra described in [6]:

$$C \geq B \oslash A \quad \Leftrightarrow \quad B \leq A \otimes C. \tag{9}$$

Using this relation and formula (2), we obtain a lower bound for the maximum service curve:

$$\gamma' \leq A^* \oslash A, \tag{10}$$

where A and A^* are the input and output cumulative flows, respectively.

But the maximum service curve is often not enough to analyze the system. For example, the minimum service curve (5) is required to calculate the maximum system delay and the maximum buffer size. Algorithms for calculating the minimum service curve of general systems are unknown to the authors.

The approach to calculating the minimum service curve proposed below involves the following "weaker" property of the min-plus convolution and deconvolution.

Property. *If* $C \leq B \bar{\oslash} A$, *then*

$$B \geq A \otimes C. \tag{11}$$

Proof.

Let $C(s) \leq (B \bar{\oslash} A)(s)$ for $s \in \mathbb{R}$. This means that for any $v \geq 0$,

$$B(s+v) - A(v) \geq \inf_{u \geq 0} \left(B(s+u) - A(u) \right) \geq C(s),$$

i.e.,

$$B(s+v) \geq C(s) + A(v). \tag{12}$$

Introducing the notation

$$t = s + v,$$

we rewrite inequality (12) as

$$B(t) \geq A(t-s) + C(s). \tag{13}$$

Inequality (13) holds for any s such that $t \geq s \geq 0$. Hence, it will be true in the limit case (for the lower bound of its right-hand side):

$$B \geq A \otimes C, \ , \ \forall t \geq 0.$$

The proof of this property is complete. ♦

Now we estimate the minimum service curve. Let A and A^* be input and output cumulative flows, respectively. Due to the property established above, the function

$$\beta' = A^* \bar{\oslash} A \tag{14}$$

satisfies the inequality $A^* \geq A \otimes \beta'$. In other words, the estimate β' is the minimum service curve.

Since the property (11) is only a necessary condition, the estimate of the minimum service curve given by (14) can lie above or below the real minimum service curve of the system. Comparing the expressions (9) and (11), we also note that $\beta \leq \gamma'$. In other words, the minimum service curve is bounded above by the maximum service curve.

Consider a special case when the system has no maximum service curve: there is an "instantaneous" processing mode for input flows. In this case, an exact value of the minimum service curve can be obtained by replacing the input and output cumulative flows with their envelopes. To do this, assume that α and α^* are the envelopes of the input and output flows, respectively. As is known, $\alpha^* = (\alpha \otimes \gamma) \oslash \beta$; see the book [6], p. 34. For $\gamma(t) = \delta_0$, this equation can be written as

$$\alpha^* = \alpha \oslash \beta .$$

Due to the property of the operator $\oslash$ (see [6], p. 123) and the minimum service curve β,

$$\alpha = \beta \otimes \alpha^*.$$

Using the commutativity of the operator $\otimes$ and the same property reversely, we arrive at the following estimate of the minimum service curve:

$$\beta' = \alpha^* \oslash \alpha.$$

If the service curves can be described by affine functions, then there exist fast convolution and deconvolution algorithms necessary for calculating the system's parameters [11]. As shown in [21], the service curves can be approximated by affine functions like the flow envelope using similar support vector algorithms.

4. MODELING AND ESTIMATING TIME CHARACTERISTICS OF CONTROL SYSTEMS

Let us describe the typical CS (Fig. 1) using the Network Calculus model. In addition, assume that the CS has redundant computing power. Under this assumption, the system can be decomposed, and each logical channel can be considered separately. Otherwise, it is necessary to examine the mutual influence of different processing channels, e.g., using a task scheduler model [11].

In the CS shown in Fig. 1, consider a separate control channel (Fig. 2). Each CS component in the channel model is characterized by its maximum and minimum service curves. According to the definitions of the minimum (5) and maximum (6) service curves, the derivation and final equations for the maximum service curve of the CS will be similar to those for the minimum service one. They can be written by simply renaming the variables and reverting the inequality signs in the relations. Therefore, all the main conclusions and considerations in this section will concern the minimum service curve β. To indicate a particular component, we will add an appropriate alphanumeric subscript to β according to the notation in Fig. 2. Level 1 of the system receives an input flow designated by an uppercase letter with an index. The input and output flows of each component will be denoted by A and A^*, respectively.

Due to the definition of the minimum service curve (5), for each element of the linear system, we have the expression

$$A^* \geq A \otimes \beta.$$

However, in practice, the characteristics of all CS elements (except communication channels) are nonlinear: the scale of the flow between the input and output

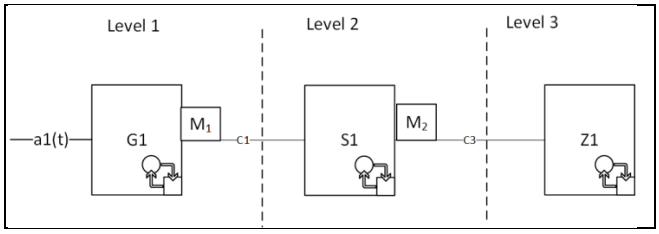


Fig. 2. Logical transmission channel *i*1 separated in CS.

changes. For example, one alarm signal at the component's input can cause a whole avalanche of related signals in the algorithms of information protection and display in the control system. As a result, the information at the component's output will increase. To incorporate the flow scale changes into the model shown in Fig. 2, we introduce a scaling function M and its inverse M^{-1}. They implement the transformations $M : A^* \to A$ and $M^{-1} : A \to A^*$, respectively, [8]. In this case, the service curve β_{Si} of system channel i with the scaling functions takes the form

$$\beta_{Si} = \beta_{Gk} \otimes M_1^{-1}(\beta_{Cn1} \otimes \beta_{Sl} \otimes M_2^{-1}(\beta_{Cn2} \otimes \beta_{Zm})), \quad (25)$$

where: $i, k, l, m \in \mathrm{N}$ are the numbers of serially connected components in the logical data processing channel at each CS level; $n1, n1 \in \mathrm{N}$ are the numbers of the communication channels used for data transmission between the components in channel $i1$; finally, M_1 and M_2 are the scaling functions of the corresponding components. The service curves β_{Cn1} and β_{Cn2} reflect the network data transmission delay T_E; the others, the data processing delay T_S in the component.

If the service curves and scaling functions can be calculated for each component, equation (15) yields bounds on the data processing delay for the entire system, depending on the input flow characteristics $ai(t)$, $i \in \mathrm{N}$. However, calculating the scaling functions M of a real system is a difficult problem not necessarily solved in practice.

To avoid difficulties with determining the scaling functions, we apply the following technique for the systems with a cyclic data processing algorithm: redefine the input and output flows and pass from the real flows to the virtual ones.

Suppose that all data received at the beginning of each cycle will be processed and transmitted to the output by the end of the cycle. Consider the function

$$\begin{cases} q(j) = \tau_j, j \in N, \\ q(0) = 0, \end{cases}$$

where j denotes the cycle number, and τ_j is the duration of cycle j. On the interval $[0, +\infty)$, we introduce the function

$$Q(x)=\sum_{l=0}^{j}q(l),\ j\le x<j+1.$$

Obviously, the step function $Q(x)$ is a flow by Definition 2.

For such a component, the output flow Q^* can be obtained from the input flow by the single-cycle shift:

$$Q^*(x)=\begin{cases}Q(x-1),x\in[1,+\infty),\\0,x\in[0,1).\end{cases}$$

Figure 3 shows the structural diagram (Fig. 2) redefined for the virtual flows in the components of types *G*, *S*, and *Z*.

For the channel with such virtual flows for the components *G*, *S*, and *Z*, we redefine the minimum and maximum service curves β and γ:

$$Q^*\ge Q\otimes\beta,$$
$$Q^*\le Q\otimes\gamma.$$

Also, we introduce the direct mappings M_0',M_1', and M_2' for the transformation $M':Q^*\to A$ and the inverse mappings for the transformation $M^{-1'}:A^*\to\mathrm{Q}$. Then the service curve for the system in Fig. 3 takes the form

$$\beta_{Si}=\beta_{Gk}\otimes M_1'\otimes\beta_{Cn1}\otimes M_1^{-1'}\otimes\otimes\beta_{Sl}\otimes M_2'\beta_{Cn2}\otimes M_2^{-1'}\otimes\beta_{Zm}. \quad (16)$$

In turn, equation (16) can be reduced to a convenient form; see subsection 5.1 of the paper [8]. For this purpose, the scaling functions M' are transferred from the component's input to the output, and the pair $(M',M^{-1'})$ at the component's output is canceled:

$$\beta_{Si}=\beta_{Gk}\otimes M_1^{-1'}(\beta_{Cn1})\otimes\beta_{Sl}\otimes\otimes M_2^{-1'}(\beta_{Cn2})\otimes\beta_{Zm}. \quad (17)$$

The partial transition from the data flows A to the cycles Q in equations (16) and (17) does not simplify the operation of scaling functions. However, under the condition

$$\beta_{Ci}\gg\beta_{\{G,S,Z\}i}, \quad (18)$$

the curve β_{Ci} can be replaced by the function

$$\delta(t)=\begin{cases}0,t=0,\\+\infty,t>0.\end{cases}$$

It is neutral with respect to min-convolution and has the property $f=\delta\otimes f$. (For example, see the book [6].)

The monotonic scaling function of the network component satisfies the relation

$$M^{-1'}(\delta(t))\to\delta(n),n\in\mathrm{N}.$$

For such a function, we may ignore β_{Ci} in equation (17), thereby eliminating the scaling functions. Physically, the assumption (18) means that the processing cycle time in the network stack corresponding to the information transmission time over the system network is negligible compared to the information processing time on a computing resource. This assumption mainly holds for modern digital control systems, where the transmitted information has a relatively small volume compared to the throughput of communication channels.

In this case, the general service curve for chain *i* of the secondary flow (see equation (17)) reduces to

$$\beta_{Si}=\beta_{Gk}\otimes\beta_{Sl}\otimes\beta_{Zm}.$$

Here $k, l, m\in\mathrm{N}$ are the numbers of the serially connected components that process data at the CS levels.

Although Network Calculus is quite "transparent" in the sense of results interpretation, it involves nonstandard characteristics of the studied objects: the flow envelope and the service curve. These characteristics are not directly measured but result from calculations. Obviously, the methods used to calculate them will also affect the reliability of the final result. Therefore, we will focus on the practical aspects of calculating the flow envelope and the service curve.

5. VALIDATION OF NETWORK CALCULUS FOR ESTIMATING TIME CHARACTERISTICS

5.1. Reference data and validation procedure

Before calculating the characteristics of the typical CS (Fig. 1), we have to validate Network Calculus on data with known statistical parameters. A test program

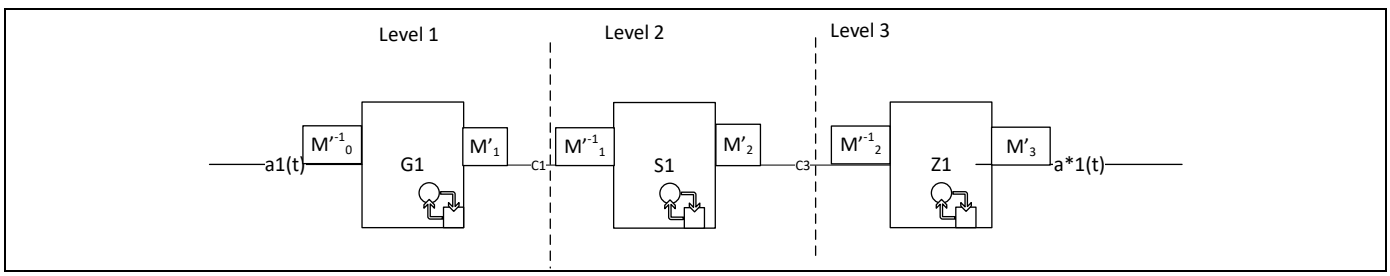


Fig. 3. The structural diagram of CS with cyclic virtual flows.

was created to simulate the CS component with a cyclic operation algorithm; the network delay *TE* and the cycle time *TS* were random variables with a given distribution. The obtained data were processed using the Network Calculus library of Matlab [10].

The subject of study was the ratio of the maximum delay determined by Network Calculus and the maximum measured delay in the sample as well as the dependence of the maximum calculated delay on the sample size and distribution.

In addition, a technique was developed to carry out measurements and obtain samples with network packet sizes and cycle times on a test program simulating the cyclic algorithm of a real system. Figure 4 shows the pseudocode of this test program. The file containing the parameters of the delay distribution function is read line by line in the loop; a random delay with a given distribution is inserted after each read operation. The duration of each cycle is recorded in the output file.

```
printTimeAndVal(InData, OutData)
 // subroutine print timestamp and input/output data counter

main
begin
   insum=0;  //  input  data flow counter
   outsum=0; ;  //  output  data flow counter



while((nch =read(par1, par2, …, parN, Distr)) >0) //read file by lines
{

  insum += nch; // increase current input  flow counter
 printTimeAndVal(insum, outsum)); //write data to file

  randn(par1, par2, …, parN, Distr);  // Distr – statistical distribution
// delay execution on random value with selected distribution
// par1, par2, …, parN - parameters of the distribution
    write(outdata);
    outsum += nch; // increase current output data   flow counter
    printTimeAndVal(fpi, insum, outsum);
}
end;
```

Fig. 4. Test program's pseudocode for generating output flows with a given delay distribution.

5.2. Testing technique and results

For each sample, three delays were calculated:

– the maximum experimental delay D_x over the entire dataset,

– the delay determined by Network Calculus according to formula (8) with the minimum service curve β estimated by formula (14),

– the delay determined by Network Calculus according to formula (8) with the lower bound (10) of the maximum service curve γ.

The calculations were carried out for samples of different sizes *L*. The data in the samples had different distributions, including those close to Gaussian and heavy-tailed ones. The numerical results presented below are rounded within 1%.

The initial test data are combined in the table available for download [22]. The data obtained with the test program are indicated by the asterisk (*).

Figure 5 shows the ratio D / D_x depending on the sample size for different distributions, where D is the maximum delay determined by Network Calculus with the service curve (14).

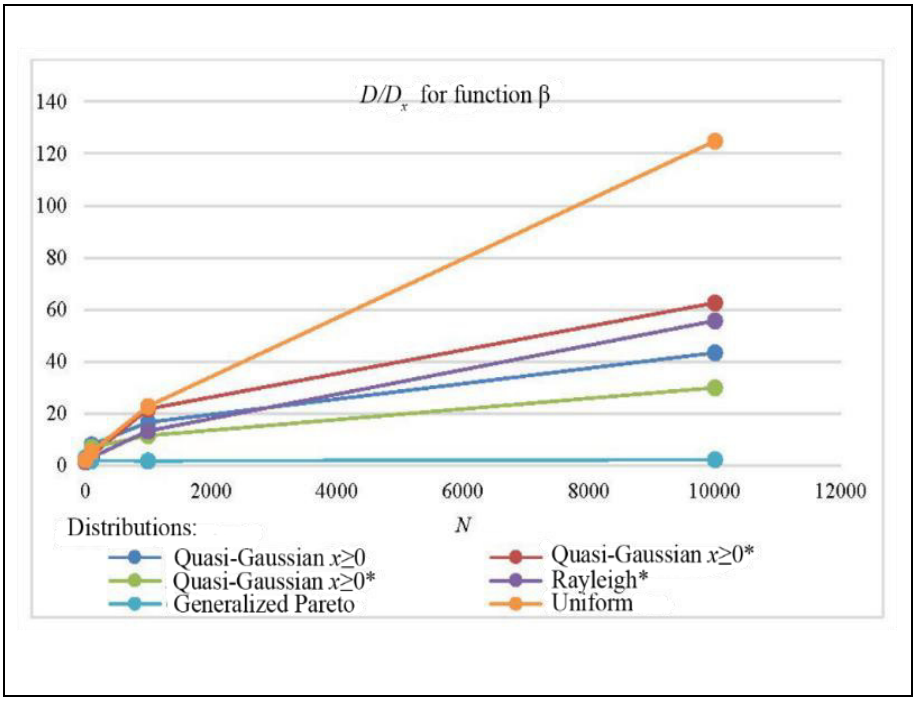


Fig. 5. Maximum delay estimate depending on sample size for β.

Figure 6 shows the ratio D / D_x depending on the sample size for different distributions, where D is the maximum delay determined by Network Calculus with the service curve (10).

The testing results allow drawing the following conclusions.

The maximum delay D estimated using the service curve (10) is close to the experimental maximum delay D_x: as a rule, the former is somewhat smaller than the latter. The resulting estimate better correlates with the real maximum delay for large samples and distributions close to Gaussian [22].

At the same time, the estimate of the maximum delay using the minimum service curve (14) is more accurate for small samples and heavy-tailed distributions. The ratio D / D_x grows with increasing the sample size, although the delay's rate of change decreases with increasing the sample size. The ratio D / D_x can reach 10^2.

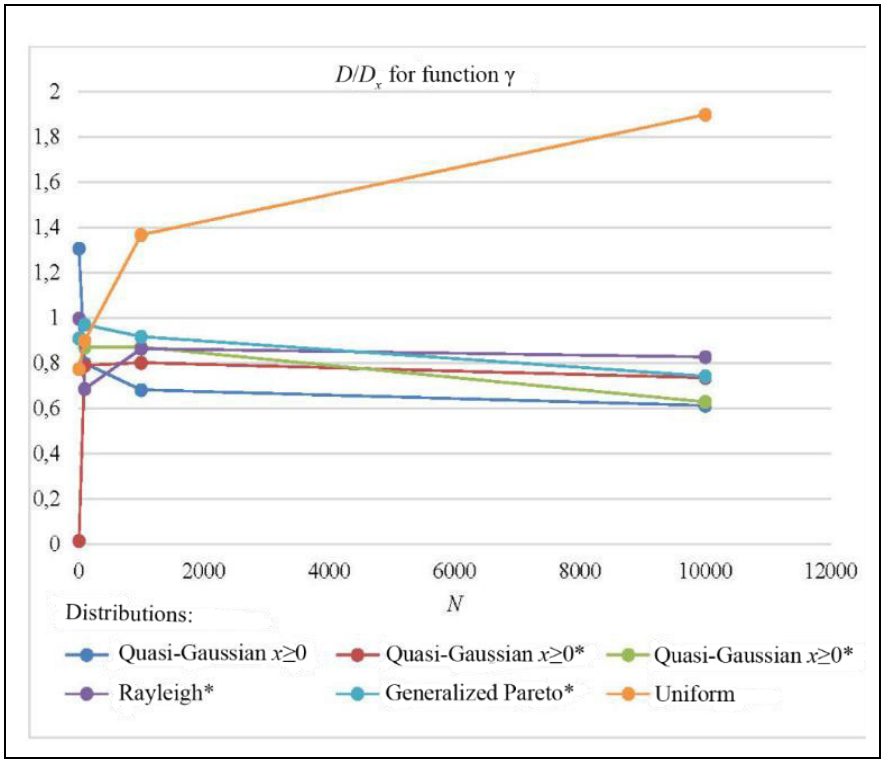


Fig. 6. Maximum delay estimate depending on sample size for γ.

Figure 7 presents the ratio D / D_x depending on the sample size and overshoot amplitude in the case of the minimum service curve.

The simulation shows that the maximum delay calculated using the service curve (10) characterizes the delay in normal operating conditions; see Fig. 6. The estimated delay is close by absolute value to the maximum delay in the sample and weakly depends on the sample size for sufficiently large samples.

Figure 8 shows typical curves yielded by Network Calculus based on the experimental data. The sample data have the Rayleigh distribution with $\sigma = 1$; however, single overshoots of 300 σ were added to the sample. For clarity, the small-sample data are given here. The upper horizontal line corresponds to the maximum delay calculated for the service curve (10). This delay is close to the maximum sample delay. The lower

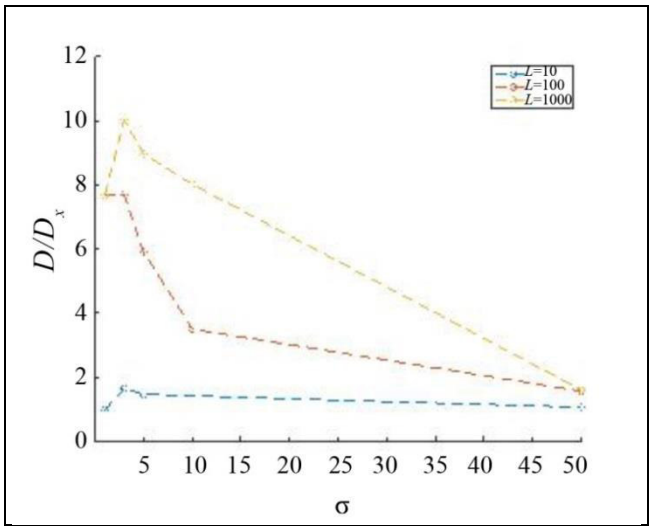


Fig. 7. The ratio of measured and calculated delays for the Rayleigh distribution with μ = 0 and σ = 300 bytes depending on single overshoot in σ and sample size *L*.

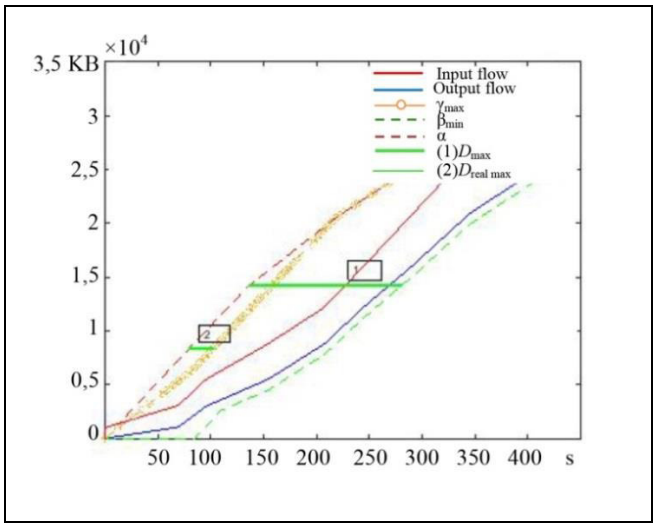


Fig. 8. Network Calculus-based experimental curves for a sample of size $L = 10$ with the Rayleigh distribution and a single overshoot of 300 σ.

horizontal line corresponds to the maximum delay estimated for the service curve (14). Clearly, the input flow envelope limits all curves on the graph from above; the estimate of the minimum service curve (14), from below.

For the samples obtained on the test example (not generated), the results turned out to be somewhat less stable. However, the differences in the order of magnitude from the same distributions obtained by direct data generation do not exceed 20%. The data acquisition procedure described above can be recommended for measurements of real systems.

5.3. Comparison of results: Network Calculus vs. classical statistical methods

According to the simulation results, the ratio D/D_x and the maximum delay depend on the distribution function of processing times, the sample size, and the number and amplitude of single overshoots in the data.

This dependence is complex due to the nonlinear formulas describing the basic Network Calculus operations (1)–(4) to find the maximum delay. According to these formulas, the flow envelope and the service curve include segments of close values arranged in descending order for the flow envelope and the maximum service curve and ascending order for the minimum service curve. (See the book [6], p. 113.)

Thus, for larger samples, the flow will contain many segments with a considerable slope of the curve; therefore, the flow envelope (7) and the service curves calculated by formulas (10) and (14) will change.

The maximum service curve (10) estimated on the sample will be similar to the flow envelope. The minimum service curve has the opposite trend (see Fig. 8). Hence, the estimate (10) depends less on variations of the input data and sample size.

Heavy-tailed distributions are characterized by some overshoots strongly differing from the rest of the values. For distributions close to Gaussian, the appearance of such overshoots in the sample is less probable, but they are characterized by a sufficient volume of data within the confidence interval. Accordingly, the general slope trend for the envelope and service curves will differ depending on the distribution of the measured parameter. The samples with single large overshoots will be characterized by the curves with a large slope value at the beginning and its subsequent sharp decrease; the samples without large overshoots, by the curves with a smooth decrease in slope (Fig. 9).

Some examples of the service curves for different sample sizes were given in the supplementary material; see Fig. 1 in [23].

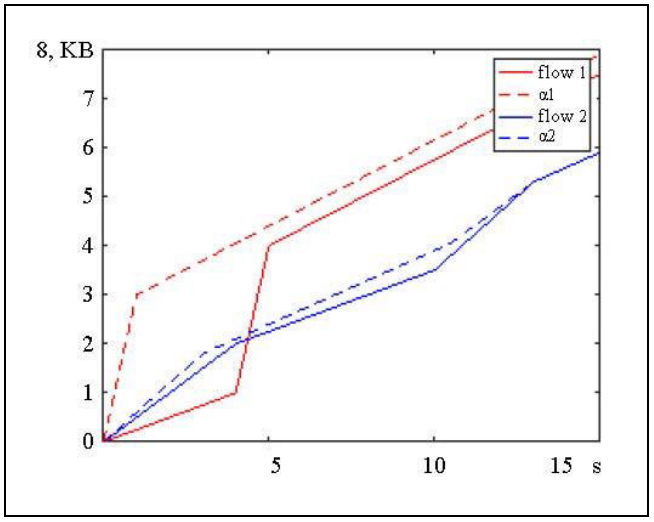


Fig. 9. Example of two cumulative flows and envelopes (α1, α2) for them. Flow 1 has overshoots in data (point 4 on axis X). Flow 2 has no overshoots in data.

These considerations explain the relationship between the delays estimated by Network Calculus and classical statistical methods. (For example, see [23].)

As is known [25], the results yielded by Network Calculus assume the worst combination of information processing conditions in the system. Graphically, this means that the segments with the greatest changes in the input flow are concentrated at the beginning of the envelope curve (the worst scenario predicted based on the observed data). For the delay based on the minimum service curve, the worst scenario is the largest data packet arriving when the server is busy and has low performance. When calculating the delay with the maximum service curve, the maximum data size corresponds to the maximum service characteristic: the maximum volume of data is accompanied by the maximum system performance, which is typical for normal operating conditions.

In both cases, the delay estimated by Network Calculus corresponds to the delay calculated using statistical methods in the scenarios described above. The probability that the delay will reach this value in a real sample corresponds to the experimental probability of this scenario. During simulations, we calculated the probability that the real delay would be less than that yielded by Network Calculus. This probability is close to 1 for the delay determined using the minimum service curve; see [22].

6. CALCULATING CONTROL SYSTEM DELAY: AN EXAMPLE

We estimated the time characteristics of the real control system described above (Fig. 1). Note that the network data transmission delays between the components were also

measured to validate the simplified formula (23) for calculating the service curve of the entire system.

Empirical distributions were calculated for the measured values, and spectral characteristics were additionally analyzed for the network delay.

Consider the maximum data processing time in a component with a cyclic operation algorithm estimated by Network Calculus.

The measurements were carried out for the components of level Z (Fig. 2). The volume of cyclic data processed is rather stable in normal operating conditions and has an average rate of change. However, under special conditions (actuation of equipment protection and lockouts or transition between modes), the volume of data and the algorithm (rate) of data processing can vary significantly.

The empirical distribution of the cycle time T_S (Fig. 10) differs from the Gaussian or Poisson and is multimode. In accordance with the algorithm of operation, each mode corresponds to a typical processing cycle for a certain class of data.

For the given sample, the estimate of maximum delay using the service curves (10), (14) was made. The following table presents the results obtained.

Simulation results for component *Z*

$L \sim 10^3$							
D_x	$p(D_x)$	D	$p(D)$	D/D_x	D'	$p(D')$	D'/D_x
0.37	~ 1	5.1	~ 1	4.9	0.32	0.87	0.3

Consider the parameters of the network data transmission delay T_E between the system components shown in Fig. 2. As an example, the data on network packets passing between the components of levels S and Z are presented. The components in this example exchange data via the TCP/IP protocol. The data transmission characteristics between other components of the system are similar.

In the experiment, the standard tcpdump utility of the OS was used for measuring the round-trip time (RTT) of a TCP packet, i.e., the period between sending the packet by the component S and receiving confirmation [26] from the component Z. The RTT measurements are shown in Fig. 11.

The typical round-trip time of a packet is tens of microseconds. However, the RTT distribution significantly differs from the Gaussian distribution inherent in physical processes or the Poisson distribution widely used in queuing theory (Fig. 12).

The distribution in Fig. 12 has three distinct periods. At the same time, no long-term periods were revealed during the experiment. The RTT spectrum analysis in Fig. 13 confirms this fact: the spectrum is noisy.

The maximum transit time of network packets is approximately 10^3 times smaller than the processing time of information in cycles. Hence, for this CS, the service curve can be calculated using the simplified formula (18).

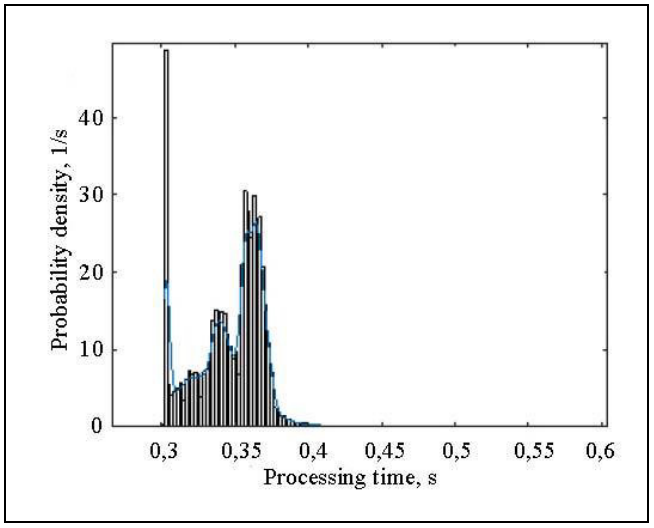


Fig. 10. The empirical probability density of cycle time for component Z. The firm line shows a smoothened envelope of the distribution.

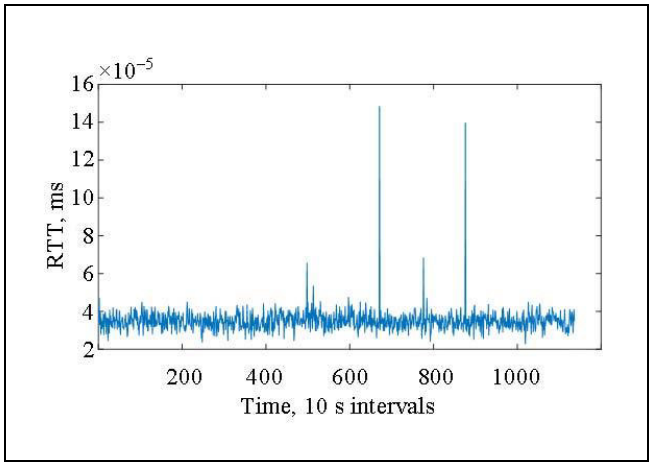


Fig. 11. RTT between the components S and Z of the real CS. The data are averaged on 10 s intervals.

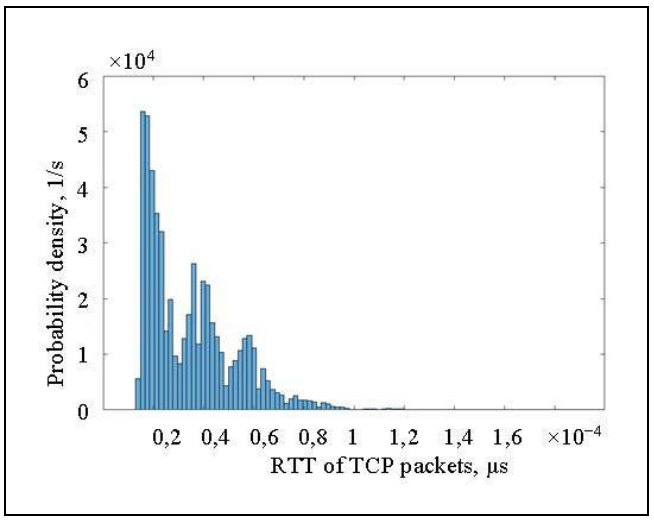


Fig. 12. The empirical probability density of RTT of TCP packets.

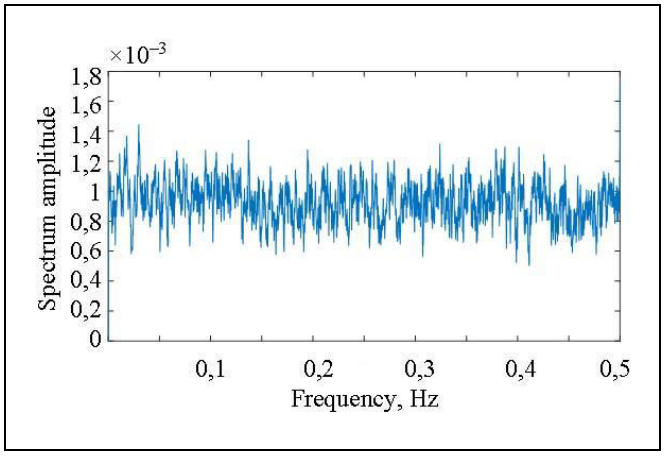


Fig. 13. The amplitude of RTT spectrum. The zero harmonic corresponding to the mean value is cut out. The data are reduced to a uniform scale with 2 s intervals between points.

In the course of measurements on the real CS, we verified that the empirical distributions (Figs. 10 and 12) have a heavy tail. The algorithm for recognizing heavy-tailed distributions [27] was employed for this purpose. It demonstrated better results than the tests based on the Kolmogorov–Smirnov criterion.

According to the real system test, the distributions of delays in the network components and the information processing components belong to the class of heavy-tailed distributions.

CONCLUSIONS. DISCUSSION OF THE RESULTS

This paper has considered the problem of validating the time characteristics of digital control systems (CSs) during their testing. CS requirements often include restrictions on the processing time of individual CS components and the information transmission time between CS components.

Constraints can be imposed on both average and maximum (limit) values of these characteristics. They are expressed either in statistical form (confidence intervals) or in standard form (admissible ranges of the absolute values) [27].

Estimating a random variable on a sample is a classical problem of mathematical statistics: it has long been developed and well described in the literature (for example, see [23]). However, the interpretation of the resulting estimates, which extends the experience of operating "ordinary" measurements of physical quantities with almost Gaussian distributions to estimating the time characteristics of digital control systems, may lead to incorrect conclusions. Let us formulate the main problems.

The procedure for validating the requirements during tests is primarily based on calculating the sample mean and sample variance (e.g., the IAEA methodology [1]). If a random variable has a finite mean and variance, the sample mean is a consistent unbiased estimate of the theoretical mean and does not depend on the type of distribution. A known disadvantage of this method is low robustness under extraneous overshoots in the sample [29]. However, sample variance, both biased and unbiased, is a consistent estimate of the variable's theoretical variance.

When interpreting the resulting estimates of the mean and variance, engineers implicitly assume that the delays obey the Gaussian law and intuitively transfer the estimated confidence intervals for the Gaussian variable to control system delays. Indeed, if a random variable has the Gaussian distribution, the sample mean and variance can be used to estimate the confidence interval for the validated parameter. However, the distribution function of delays in CSs is generally non-Gaussian.

The physical nature of the measured quantity (time) restricts the form of its distribution: at least, it is bounded on the left. If the technical requirements specify the maximum absolute value (e.g., the signal transit time between the CS components should not exceed a given threshold), this condition implies that the distribution of the random variable is also bounded on the right. Therefore, the restrictions dictate that the distribution is not, in the strict sense, the distribution of a Gaussian random variable.

According to the study of a real CS (see above), the distribution of delays in the network components and the components processing information significantly differs from the Gaussian one: it often has a multimode nature and belongs to the class of heavy-tailed distributions.

In the general case, the Chebyshev inequality yields a (very rough) estimate for the probability that a random variable will exceed a given threshold. Therefore, when estimating time characteristics, it is necessary to obtain an appropriate distribution and then apply statistical estimation methods for this class of distributions or, as an alternative, use non-statistical methods.

This paper has considered a non-statistical approach to estimating the time delay in control systems based on Network Calculus. This method is not completely new; however, it is still underinvestigated by researchers. When applying it to computer systems analysis, we should consider some of the features of the method. One feature is insufficient transparency in correlating the results yielded by Network Calculus with those yielded by classical statistical methods for estimating time parameters of control systems. In addition, the system's input data necessary for this

method are generally not specified as "passport parameters" of the system and the information processed by it. Such data include flow envelopes, service curves, scaling functions for uneven data flows, etc. The technical difficulties of Network Calculus are well known, and separate approaches have been developed to resolve them partially; for example, see [8, 10, 11, 19, 30]. However, these solutions also require initial data about the system, which are unavailable for the user or are poorly formalized. Moreover, there is no general methodology for estimating the minimum service curve, an important parameter of Network Calculus.

Therefore, this paper has proposed methods for estimating the minimum service curve using the input and output cumulative data flows. For a special case of a control system with a cyclic information processing algorithm, a simplified formula without scaling functions has been presented for calculating the system service curve.

We have investigated the correlation between the maximum delay estimated by Network Calculus with the results obtained using the statistical analysis of time delay samples. In particular, it has been established that the maximum delay in the data sample and the maximum delay estimated by Network Calculus are closest if the distribution of the sample data has single large overshoots. This property is inherent in heavy-tailed distributions. It has been hypothesized that the maximum delay relates to the probability of a rare event – the sequential arrival of a significant volume of data – under a low server performance for the minimum service curve.

The research presented above allows developing non-statistical estimation methods for time characteristics of digital control systems considering the peculiarities of their operation. Also, it significantly expands the application area of Network Calculus for estimating the parameters of control systems.

The problem of describing closed-loop paths, characteristic of control systems, goes beyond the scope of this paper. A corresponding mathematical apparatus has been developed within Network Calculus; see [6, 31]. However, it has not been properly validated for real systems.

REFERENCES

1. On-line Monitoring of Instrumentation in Research Reactors, *IAEA TECDOC Series*, no. 1830, IAEA: Vienna, 2017.
2. Golshani, G., Taylor, G., and Pisica, I., Simulation of Power System Substation Communications Architecture Based on IEC 61850 Standard, *Proceedings of the 2014 49th International Universities Power Engineering Conference* (*UPEC*), 2014, pp. 1–6. DOI: 10.1109/UPEC.2014.6934745.
3. Ahmad, Z. and Durad, M.H., Development of SCADA Simulator Using Omnet++, *Proceedings of the 2019 16th International Bhurban Conference on Applied Sciences and Technology* (*IBCAST*), 2019, pp. 676–680. DOI: 10.1109/IBCAST.2019.8667158.
4. Chen, L., Zhang, K., Xia, Y., and Hu, G., Scheme Design and Real-Time Performance Analysis of Information Communication Network Used in Substation Area Backup Protection, *Proceedings of the 2012 Power Engineering and Automation Conference*, 2012, pp. 1–4. DOI: 10.1109/PEAM.2012.6612461.
5. Zhang, Z., Huang, X., Keune, B., et al., Modeling and Simulation of Data Flow for VLAN-Based Communication in Substations, *IEEE Systems Journal*, 2017, no. 4, pp. 2467–2478. DOI: 10.1109/JSYST.2015.2428058.
6. Le Boudec, J.-Y. and Thiran, P., *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*, Springer Verlag, 2019.
7. Schioler. H., Dalsgaard. J., Larsen. K., and Jessen, J., CyNC – a Method for Real Time Analysis of Systems with Cyclic Data Flows, *J. Embed. Comput.*, 2006, vol. 2, no. 3-4, pp. 347–360.
8. Fidler, M. and Schmitt, J., On the Way to a Distributed Systems Calculus: An End-to-End Network Calculus with Data Scaling, *SIGMETRICS Perform. Eval. Rev.*, 2006, vol. 34, no. 1, pp. 287–298. DOI: 10.1145/1140277.1140310.
9. Promyslov, V.G., Simulation of Computer System with a Flow Alternation between Components, *Control Sciences*, 2012, no. 1, pp. 62–70. (In Russian.)
10. Bouillard, A. and Thierry, É., An Algorithmic Toolbox for Network Calculus, *Discrete Event Dyn. Syst.*, 2008, vol. 18, pp. 3–49. DOI: 10.1007/s10626-007-0028-x.
11. Bouillard, A., Jouhet, L., and Thierry, E., Comparison of Different Classes of Service Curves in Network Calculus, *Proceedings of the 10th International Workshop on Discrete Event Systems* (*WODES*), 2010, pp. 316–321.
12. Byvaikov, M.E., Zharko, E.F., Mengazetdinov, N.E., et al., Experience from Design and Application of the Top-Level System of the Process Control System of Nuclear Power-Plant, *Automation and Remote Control*, 2006, vol. 67, no. 5, pp. 735–747.
13. Ibrahim, W.Z. and Sallam, H., Instrumentation and Controls Architectures in New NPPs, *Int. J. of Nuclear Knowledge Management*, 2014, vol. 6, pp. 283–302. DOI: 10.1504/IJNKM.2014.062830.
14. Cruz, R.L., A Calculus for Network Delay. Part I: Network Elements in Isolation, *IEEE Trans. on Information Theory*, 1991, vol. 37, pp. 114–131.
15. Cruz, R.L., A Calculus for Network Delay. Part II: Network Analysis Information Theory, *IEEE Trans. on Information Theory*, 1991, vol. 37, pp. 132–141.
16. Baccelli, F., Cohen, G., Olsder, G.J., and Quadrat, J.-P., *Synchronization and Linearity: An Algebra for Discrete Event Systems*, New York: John Wiley & Sons, 1992.
17. Ahmadi, H., Martí, J. R., and Moshref, A., Piecewise Linear Approximation of Generators Cost Functions Using Max-Affine Functions, *Proceedings of the 2013 IEEE Power Energy Society General Meeting*, 2013, pp. 1–5. DOI: 10.1109/PESMG.2013.6672353.
18. Camponogara, E. and Nazari, L., Models and Algorithms for Optimal Piecewise-Linear Function Approximation, *Mathematical Problems in Engineering*, 2015, vol. 2015, article no. 876862. DOI: 0.1155/2015/876862.
19. Baybulatov, A.A. and Promyslov, V.G., The Approximation of Envelope in Network Calculus Applications, *Control Sciences*, 2016, no. 6, pp. 59–64. (In Russian.)

20. Vapnik, V.N., *Statistical Learning Theory*, New York: John Wiley, 1998.
21. Baybulatov, A.A. and Promyslov, V.G., Control System Availability Assessment via Maximum Delay Calculation, *Proceedings of the 2019 International Conference on Industrial Engineering, Applications and Manufacturing* (*ICIEAM*), Sochi: IEEE, 2019, pp. 1–6. DOI: 10.1109/ICIEAM.2019.8743012.
22. URL: https://www.dropbox.com/s/zhohii8d9qfqrki/widetable2ru.xlsx?dl=0.
23. URL: https://www.dropbox.com/s/yurroijpayx6p38/Table2en.docx?dl=0
24. Cramér, H., *Mathematical Methods of Statistics*, Princeton: Princeton University Press, 1999. Originally published in 1946.
25. Fidler, M. and Rizk, A., A Guide to the Stochastic Network Calculus, *IEEE Communications Surveys & Tutorials*, 2015, vol. 17, no. 1. DOI: 10.1109/COMST.2014.2337060.
26. RFC 793. Transmission Control Protocol. Protocol Specification, 1981. https://tools.ietf.org/html/rfc793
27. Burnecki, K., Wylomanska, A., and Chechkin, A., Discriminating between Light- and Heavy-Tailed Distributions with Limit Theorem, *PLoS ONE*, 2015, vol. 10(12), article no. e0145604. DOI: 10.1371/journal.pone.0145604.
28. *GOST* (State Standard) *28195-89*: *Software Quality Assessment. General Regulations*, 1989.
29. Smith, M., *Statistical Analysis Handbook. A Comprehensive Handbook of Statistical Concepts, Techniques and Software Tools*, Edinburgh: The Winchelsea Press, 2018.
30. Promyslov, V., Using the Method of Network Calculus to Simulate the Computerized Control System with Non-Uniform Data Flow, *IFAC-PapersOnline*, 2012, vol. 45, no. 6, pp. 645–648. DOI: 10.3182/20120523-3-RO-2023.00273.
31. Baybulatov, A. and Promyslov, V., On a Deterministic Approach to Solving Industrial Control System Problems, *Proceedings of the 2020 International Russian Automation Conference* (*RusAutoCon*), Sochi, 2020, pp. 115–120. DOI: 10.1109/RusAutoCon49822.2020.9208149.

This paper was recommended for publication by V.M. Vishnevsky, a member of the Editorial Board.

Received March 23, 2021, and revised June 15, 2021.
Accepted June 22, 2021.

Author information

Promyslov, Vitalii Georgievich. Cand. Sci. (Phys.-Math.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia, ✉ v1925@mail.ru

Semenkov, Kirill Valer'evich. Cand. Sci. (Phys.-Math.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia, ✉ semenkovk@mail.ru

Cite this article

Promyslov, V.G. and Semenkov, K.V., Estimating Time Characteristics of Control Systems with Cyclic Operation: A Network Calculus Approach. *Control Sciences* **4,** 41–53 (2021). http://doi.org/10.25728/cs.2021.4.5

Original Russian Text © Promyslov, V.G., Semenkov, K.V., 2021, published in *Problemy Upravleniya*, 2021, no. 4, pp. 50–65.

DOI: http://doi.org/10.25728/cs.2021.4.6

USING PIECEWISE FUNCTIONS TO NORMALIZE INPUT VARIABLES OF FUZZY INFERENCE SYSTEMS

A.A. Sorokin

Astrakhan State Technical University, Astrakhan, Russia

✉ alsorokin.astu@mail.ru

Abstract. This paper proposes a method for normalizing the input variables of fuzzy inference systems (FISs), which are used in assessing integrally the state of a complex object. The method involves piecewise functions: the variable's range is divided into several intervals (the length of each interval depends the variable's specifics), and a particular function is assigned to each interval. This function shows the patterns of the variable's variations on the normalized scale relative to its variations on the absolute scale. The set of these functions for the entire range of the variable forms the normalization operator. When implementing the normalization operator, the functions are selected so that after transformation, all input variables positively correlate with the output variable. This approach simplifies the construction of FISs: the same terms of the input variables have the same semantic meaning after transformation. According to the simulation results, FISs with the proposed normalization method are adequate to similar FISs without the normalization of the input variables. The proposed normalization method allows reducing the number of rules in the FIS knowledge base if the input variables have an optimum of their influence on the value of the output variable.

Keywords: fuzzy inference system, normalization, input variable, knowledge base, rule, integral assessment, information processing.

INTRODUCTION

Among other tasks, decision support systems are developed for integral assessment to rank a definite group of objects. Ranking serves for various purposes, e.g., to assess the creditworthiness of bank clients. Analysis of the publications [1–6] shows that fuzzy inference systems (FISs) are widely used to solve such problems.

According to [1–9], many FISs aggregate parameters with different measurement units, different ranges of assessment scales, different influences on the output variable, and different correlations with the values of the output variable. Therefore, to simplify the design of FISs, normalization methods were adopted in [1, 2, 7–12]: the values of input variables were reduced to a single scale. As demonstrated therein, such methods allow using identical membership functions (MFs) when describing the input variables and making the FIS invariant to changing the absolute value range of the input variables: if necessary, the normalization operator is subject to changes. Considering the results of [1, 2, 7–13], we divide all normalization methods into two classes:

– Class 1 contains the methods in which a mathematical function is used for transformation, and the normalized parameter interacts with the constants characterizing the normalized sample of values [1, 7–13].

– Class 2 contains the methods in which an interval of the normalized values is assigned to an interval of the initial parameter values [2].

Among the restrictions of class 1 methods, we mention the difficulty of reducing the absolute values of the input variable with a nonlinear influence on the output variable to the normalized values with a linear influence on the FIS final value. Class 2 methods have restrictions as well: in the publications discussed above, the mathematical methods for transforming the absolute values of variables into the normalized ones on a given interval were not formally implemented; the transformation itself preserved the correlation with the output variable. Such restrictions complicate the design of rules to transform the values of the input parameters into the value of the output variable in FISs.

Thus, this paper aims at improving the methods for normalizing the values of input variables in fuzzy inference systems.

1. A SURVEY OF RELATED WORKS AND PROBLEM STATEMENT

As described in [1, 2, 14], an FIS is implemented in the following stages:

- forming an array of aggregated variables $X = \{x_i\} : i = \overline{1, n}$, where each variable x_i has a scale sc_i;
- forming an array of output variables $Y = \{y_j\} : j = \overline{1, m}$.

Below, we will consider FISs with one output variable only.

After the array of the input variables x_i is formed, an array of terms $T_i = \{t_k^{x_i}\} : k = \overline{1, r}$ is specified for each variable. Each term $t_k^{x_i}$ has a semantic name characterizing the state of the described parameter x_i. For each term $t_k^{x_i}$, an MF $\mu_k^{x_i}(x_i)$ is formed, where $\mu_k^{x_i}$ denotes an operator for transforming an input variable from a crisp value x_i to a fuzzy one $x_{fuzz,i}$. The MF value shows the level of confidence that a crisp value of a variable corresponds to a specific term. The MP domain is the interval [0, 1]: unity indicates that the value of the variable fully matches the semantic meaning of the term, and zero indicates that the value of the variable does not match the semantic meaning of the term. Triangular and trapezoidal functions are often used to form MFs.

The variable aggregation operator is implemented by forming rules. Often a rule is a logical statement written as (*if* $A \Rightarrow B$, ξ, where A denotes the set of initial conditions (antecedent), B is a conclusion (consequent), and ξ gives the coefficient of confidence in the rule). (Further, let $\xi = 1$). When forming the set A, all value combinations for the terms of the input variables are enumerated, which form definite values of the terms of the output variables B. There are two structures of rules: Multiple Input Single Output (MISO) and Multiple Input Multiple Output (MIMO). Various logical operations "and," "or," and "not" can be used to aggregate input variables. This paper will study FISs based on MISO rules with the "and" operator to aggregate input variables. We form a knowledge base (KB) by enumerating all term combinations for the array of input variables and specifying the target value of the output variables. It displays the pattern of the influence of the input variables values on the output variables. Generally, an FIS with MISO rules can be described by

$$y = F_{\text{FIS}}(x_1, \ldots, x_i),$$

where F_{FIS} denotes a fuzzy inference operator for the variables array $X = \{x_i\}$. Figure 1*a* shows an example of an FIS aggregating the variables x_1 and x_2. According to the monograph [1], variables can be supplied to the FIS input in absolute or normalized measurement units. In the latter case, the values measured on a single scale are supplied to the FIS input. The transition from absolute values x_{abs} to the normalized units is performed using a normalization operator f_{norm}:

$$x_{\text{norm}} = f_{\text{norm}}(x_{\text{abs}}).$$

Figure 1*b* presents the FIS aggregating the variables x_1 and x_2 subjected to normalization [1]. Here, the normalizer block implements the operator f_{norm}.

Analysis of the publications [3–6, 14, 15] shows that the input variables often have the following influences on the integral assessment of the object's state:

– directly proportional (e.g., the income of a borrower who wants to get a loan from a bank: the input variable positively correlated with the output value characterizing the reasonability of issuing a loan);

– inversely proportional (e.g., the debt load of a bank client applying for the next loan: the input variable negatively correlates with the output value);

– optimal: there is an interval of input parameter values (or even a point) beyond which, on the left or right, the value of the final assessment becomes worse (e.g., the borrower's age).

Additionally, the aggregated variables can have different measurement units, different ranges of values, and different levels of influence on the final result. In turn, their degrees of significance can depend on the variable value. If the rules are formed using expertise,

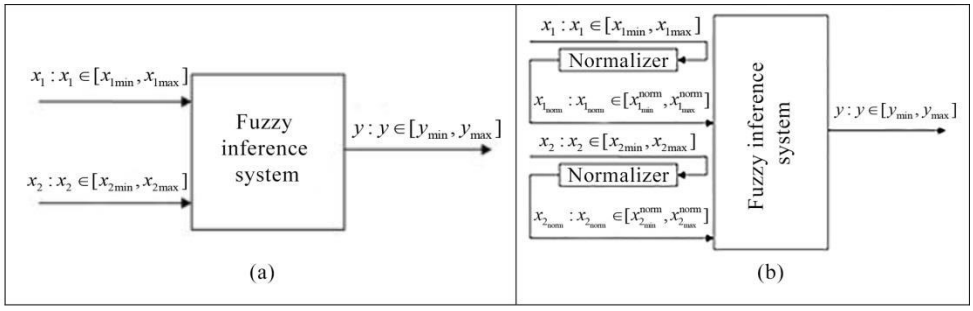


Fig. 1. The structure of models under study.

such a variety of properties of variables may cause technical errors in logical inference. To reduce the variety of properties, the authors [1, 2, 8, 9] proposed normalizing the variables to the interval [0, 1]. Considering the results of [1, 2, 7–13], we identify normalization methods based on the construction of a general pattern for transforming the variables (class 1 methods) or dividing the absolute value range of the variables into intervals and then assigning intervals on a normalized scale to each original interval (class 2 methods). When applying class 1 methods, the maximum and minimum values of the parameter and statistical indicators of the normalized sample can be considered. The papers [7, 10] described normalization methods based on the relation

$$x_{\text{norm}} = x_{\text{abs}} / x_{\max}^{\text{abs}}, \quad (1)$$

where $x_{\max}^{\text{abs}}$ is the parameter's maximum value in absolute units. The authors [1, 11] proposed to consider the minimum value $x_{\min}^{\text{abs}}$:

$$x_{\text{norm}} = \left(x_{\text{abs}} - x_{\min}^{\text{abs}}\right) / \left(x_{\max}^{\text{abs}} - x_{\min}^{\text{abs}}\right), \quad (2)$$

or

$$x_{\text{norm}} = \left(x_{\text{abs}} - x_{\max}^{\text{abs}}\right) / \left(x_{\max}^{\text{abs}} - x_{\min}^{\text{abs}}\right). \quad (3)$$

The following modifications of the relations (2), (3) were presented in [12]:

$$x_{\text{norm}} = D_1 \frac{x_{\text{abs}} - x_{\min}^{\text{abs}}}{x_{\max}^{\text{abs}} - x_{\min}^{\text{abs}}} + D_2, \quad (4)$$

where D_1 and D_2 denote some constants. The paper [13] introduced a method for normalizing input variables with the operator

$$x_{\text{norm}} = \left(x_{\text{abs}} - \overline{x}\right) / \sigma, \quad (5)$$

where $\overline{x}$ and σ are the sample mean and the standard deviation in the normalized sample, respectively. Note that with the operator (5), the range of x_{norm} does not fall into [0, 1], and the variable's range depends on its values in absolute measurement units. This feature may cause additional difficulties when implementing FISs. According to the authors cited, such a relation is appropriate for transforming the data used to train fuzzy neural networks. The monograph [1] proposed methods based on the relations

$$x_{\text{norm}} = \left(x_{\text{abs}} - x_{\text{mean}}^{\text{abs}}\right) / \left(x_{\max}^{\text{abs}} - x_{\min}^{\text{abs}}\right), \quad (6)$$

where $x_{\text{mean}} = 0.5\left(x_{\max}^{\text{abs}} + x_{\min}^{\text{abs}}\right)$.

As noted in [1], the operators (1)–(4), (6) can be generalized to the straight-line equation

$$x_{\text{norm}} = k x_{\text{abs}} + b, \quad (7)$$

where k and b denote some constants.

Note that the methods described by (1)–(7) have the following common feature: it is difficult to consider the influence of the normalized parameter on the value of the output variable depending on the value of this parameter. Also, these methods do not yield the optimum of the influence of the input parameter value on the output variable if it lies between the maximum and minimum absolute values of the variable. The nonlinear influence can be taken into account using class 2 methods. In the monograph [2], certain intervals on the normalized scale were assigned to intervals on the absolute scale of the parameter. Often, the same number of intervals are used relative to the middle of the normalization axis. To illustrate this method, Table 1 shows an example where the parameter $x_{\text{abs}} \in [x_0^{\text{abs}}, x_5^{\text{abs}}]$ is normalized to the interval $\left[x_0^{\text{norm}}, x_5^{\text{norm}}\right]$.

Table 1

Variable normalization: example

Absolute value interval	Normalized value interval	Interval no.
$\left[x_0^{\text{abs}}, x_1^{\text{abs}}\right]$	$\left[x_0^{\text{norm}}, x_1^{\text{norm}}\right]$	–2
$\left[x_1^{\text{abs}}, x_2^{\text{abs}}\right]$	$\left[x_1^{\text{norm}}, x_2^{\text{norm}}\right]$	–1
$\left[x_2^{\text{abs}}, x_3^{\text{abs}}\right]$	$\left[x_2^{\text{norm}}, x_3^{\text{norm}}\right]$	0
$\left[x_3^{\text{abs}}, x_4^{\text{abs}}\right]$	$\left[x_3^{\text{norm}}, x_4^{\text{norm}}\right]$	1
$\left[x_4^{\text{abs}}, x_5^{\text{abs}}\right]$	$\left[x_4^{\text{norm}}, x_5^{\text{norm}}\right]$	2

According to the monograph [2], the normalization operator ensures the conditions $x_0^{\text{abs}} = x_{\min}^{\text{abs}}$, $x_5^{\text{abs}} = x_{\max}^{\text{abs}}$, $x_0^{\text{norm}} = x_{\min}^{\text{norm}}$, and $x_5^{\text{norm}} = x_{\max}^{\text{norm}}$: there is a positive correlation of the variables. Thus, the correlation between the input and output variables does not change after the normalization described above. In addition, the monograph [2] suggested no analytical methods for transforming absolute values into the normalized ones; provided no recommendations on choosing the number of intervals to divide the axis of normalized values and the limits of these intervals; gave no explanations on the influence of input variable normalization on the accuracy of the resulting FIS compared to the original FIS (in which the variables are measured in absolute units).

The method described in [16, 17], originally not designed for FISs, can be considered a development of the normalization procedure suggested in [2]. This method divides the normalized values of a variable into intervals with the indication of points (their limits) and assigns to the points (the limits of the normalized-scale intervals) some values of the variable on the absolute scale. The resulting pairs of points are used to construct a curve, implementing the transformation of a variable from one scale to another. The paper [17]

proposed selecting points to divide the interval of normalized values into parts: $0 \sim x_{\text{norm}}^{0}$, $0.25 \sim x_{\text{norm}}^{0.25}$, $0.5 \sim x_{\text{norm}}^{0.5}$, $0.75 \sim x_{\text{norm}}^{0.75}$, and $1 \sim x_{\text{norm}}^{1}$. Some values of the variable on the absolute scale are assigned to them: x_{abs}^{0}, $x_{\text{abs}}^{0.25}$, $x_{\text{abs}}^{0.5}$, $x_{\text{abs}}^{0.75}$, and x_{abs}^{1}. The resulting transformation operator is represented by a broken line constructed from the pairs of points $\left(x_{\text{abs}}^{0}; x_{\text{norm}}^{0}\right)$, $\left(x_{\text{abs}}^{0.25}; x_{\text{norm}}^{0.25}\right)$, $\left(x_{\text{abs}}^{0.5}; x_{\text{norm}}^{0.5}\right)$, $\left(x_{\text{abs}}^{0.75}; x_{\text{norm}}^{0.75}\right)$, and $\left(x_{\text{abs}}^{1}; x_{\text{norm}}^{1}\right)$. However, the papers [16, 17] described no analytical methods for transforming the absolute values into the normalized ones. Moreover, similar to the method described in [2], the values positively correlate before and after transformation: the type of correlation between the input and output variables remains the same.

Thus, the methods for constructing FISs [1, 2, 14] and the ones for normalizing input variables [1, 2, 8–13] have several restrictions. They relate to the properties of the input variables aggregated using FISs. Among such properties of input variables, we mention different measurement units, different ranges of values, different and (or) uneven levels of influence on the output variable, and different types of correlation with the output variable. Due to such properties of the variables, experts face difficulties when forming rules for the FIS knowledge base.

Some of the restrictions are eliminated using normalization methods. Such methods mainly transform various values of the input variables to a single scale. The disadvantages of the normalization methods often proceed from statistical indicators of the normalized samples of input variables values (usually, the maximum, minimum, and average values). The most universal properties of variable normalization are observed for a method that assigns an interval on the normalized scale to intervals on the absolute values of the parameter. Therefore, we will improve this method below.

2. IMPROVING THE NORMALIZATION METHOD FOR INPUT VARIABLES OF FUZZY INFERENCE SYSTEMS

2.1 Proposed modifications

To improve the approach described in the monograph [2], we propose a normalization method based on an operator f_{norm} for transforming the absolute values of an input variable into the normalized ones so that (1) the value of the output variable will not decrease with an increase in the normalized value of the input variable and (2) the value of the output variable will not increase with a decrease in the normalized value of the input variable. Assume that trapezoidal MFs are used to describe the input variables. Such an assumption is well-grounded: according to [1, 2, 14], this type of MFs is widespread in applications, and the corresponding MFs have crisp limits of the support and the core. (The crisp limits of the core and the support of MFs will be used below.)

Within the proposed method, the normalization operator f_{norm} is implemented using a set of functions: a certain pattern $f_{\text{norm},z}$ is assigned to each zth interval of the absolute values x_{abs} of an input variable. In other words,

$$x_{\text{norm}} = \begin{cases} f_{\text{norm},1}\left(x_{\text{abs}}\right) \text{if } x_{\text{abs}} \in \left[x_{\text{abs},i_{\min}}, x_{\text{abs}.i_1} \right), \\ f_{\text{norm},2}\left(x_{\text{abs}}\right) \text{if } x_{\text{abs}} \in \left[x_{\text{abs}.i_1}, x_{\text{abs}.i_2} \right), \\ \quad \ldots \\ f_{\text{norm},z}\left(x_{\text{abs}}\right) \text{if } x_{\text{abs}} \in \left[x_{\text{abs}.i_{\max}-1}, x_{\text{abs}.i_{\max}} \right). \end{cases} \quad (8)$$

If the patterns $f_{\text{norm},z}$ for neighbor intervals are the same, the intervals are combined. The set of functions (8) is formed so that the minimum and maximum values of the normalized variable, $x_{\text{norm}}^{\min}$ and $x_{\text{norm}}^{\min}$, have the worst (maximum negative) and best (maximum positive), respectively, influences on the value of the output variable of the FIS. We propose implementing the operators $f_{\text{norm},z}$ as follows:

1. Define the maximum and minimum absolute value of the variable $x_{\text{abs}.i}$.

2. Define the maximum $x_{\text{norm}}^{\max}$ and minimum $x_{\text{norm}}^{\min}$ values of this variable in relative units (in this paper, $x_{\text{norm}}^{\min} = 0$ and $x_{\text{norm}}^{\max} = 100$).

3. On the absolute scale, define the interval limits z (for the set of functions (8), the points $x_{\text{abs}.i_1}$, $x_{\text{abs}.i_2}$,…, $x_{\text{abs}.i_{\max}-1}$); on the relative scale, define their counterparts (the point x_{norm,i_1} corresponds to the point $x_{\text{abs}.i_1}$, etc.) in the following way:

3.1. Considering the recommendations of [1], experts give the primary representation of a fuzzy input variable in the traditional form using trapezoidal MFs (in this paper, triangular MFs are studied as a special case of trapezoidal MFs: the core degenerates into a point).

3.2. The interval limits are the limits of the MF cores of the variable's primary fuzzy representation.

3.3. Similar types of MFs are used to describe the variables after normalization. The interval limits are the points corresponding to the limits of the MF cores of the variable's primary fuzzy representation.

3.4. If the core of the term's MF belongs to an interval with the optimality domain, it is divided into two equal intervals.

4. For each interval of z, define the function $f_{\text{norm},z}\left(x_{\text{abs}.i}\right) = x_{\text{norm}.i}$.

5. Based on the set of functions $f_{\text{norm},z}$, construct the normalization operator (8).

Following the monograph [1], we propose implementing the functions $f_{\text{norm},z}$ as straight lines passing through two points: the limits of the intervals on the normalized and absolute scales. For example, let points 1 and 2 for constructing the operator $f_{\text{norm,l}}$ have the coordinates $\left(x_{\text{abs},i_{\min}}, x^{\min}_{\text{norm}}\right)$ and $\left(x_{\text{abs},i_1}, x_{\text{norm},i_1}\right)$, respectively. Then the operator is given by

$$f_{\text{norm,l}}\left(x_{\text{abs}}\right) = \left[\left(x^{\text{norm}}_{i_1} - x^{\text{norm}}_{i_{\min}}\right)\times \right.$$
$$\left.\times\left(x_{i_{\text{abs}}} - x_{i_{\min}}\right)/\left(x_{i_1} - x_{i_{\min}}\right)\right] + x^{\text{norm}}_{i_{\min}}. \quad (9)$$

Figure 2*a* shows the normalization operator (8) implemented for the input variables positively correlating with the values of the output variable. The case of their negative correlation is illustrated in Fig. 2*b*. Finally, Fig. 2*c* corresponds to the situation in which the input variables have an optimum of their influence on the value of the output variable.

The proposed modifications allow:

– determining the number of intervals for dividing the normalized value axis based on the methods used to construct the MF of the FIS (it equals the number of pairs of the MF core limits if normalization applies to the variables positively or negatively correlating with the output variable, or plus 1 if one optimum affects the value of the output variable);

– ensuring that after normalization, the input variables have the same correlation with the output variable;

– considering changes in the degree of significance of the input variable's influence on the output value of the FIS, depending on the variable's value.

2.2. Verification of the proposed modifications

The proposed modifications of the normalization method were verified by mathematical modeling. Several FISs based on the zero-order Takagi–Sugeno model were constructed. According to [18], a feature of such models is using constants to describe the MF of the output variable. The FISs under study had two input variables, x_1 and x_2, and one output variable y. Six experiments were carried out:

– The variables x_1 and x_2 positively correlate with the variable y.

– The variables x_1 and x_2 negatively correlate with the variable y.

– The variables x_1 and x_2 have an optimum with respect to the variable y.

– The variable x_1 has a positive correlation with the variable y; the variable x_2, a negative correlation with the variable y.

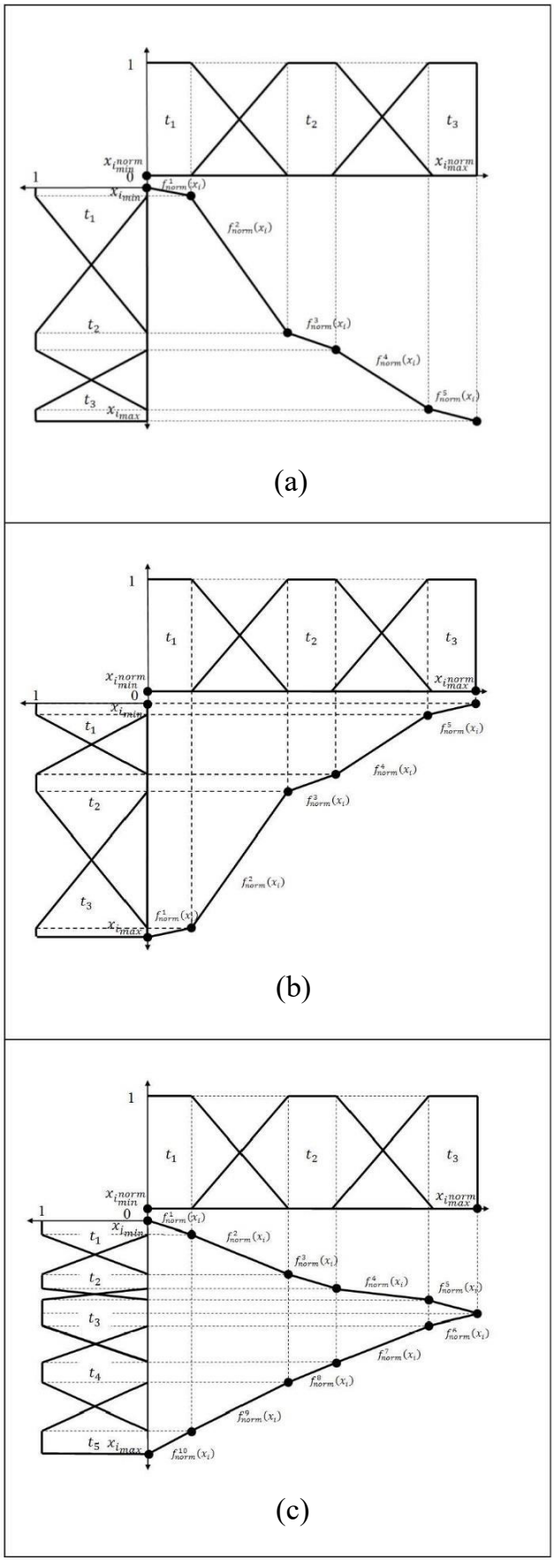


Fig. 2. Normalization operators for input variables of FISs: some examples.

– The variable x_1 has a positive correlation with the variable y; the variable x_2, an optimum with respect to the variable y.

– The variable x_1 has a negative correlation with the variable y; the variable x_2, an optimum with respect to the variable y.

Within each experiment, two FISs were investigated: the FIS of type 1 (the variables x_1 and x_2 normalized by the proposed method) and the FIS of type 2 (the variables x_1 and x_2 processed without normalization). The absolute values of the variables x_1 and x_2 were measured in the range from 0 to 10 points; the values of the variables x_1 and x_2 after normalization, in the range from 0 to 100 points; the value of the variable y, in the range from 0 to 100 points.

If a variable passed normalization, its MFs are shown in Fig. 3*a*. If a variable positively correlated with the variable y, its MFs are shown in Fig. 3*b*. If a variable negatively correlated with the variable y, its MFs are shown in Fig. 3*c*. If a variable had an optimum with respect to the variable y, its MFs are shown in Fig. 3*d* (variant 1) and Fig. 3*e* (variant 2).

The terms in Fig. 3 are described as follows. In Fig. 3*a*: *NB* indicates a worst influence on the output variable; *Z*, a medium influence on the output variable; *PB*, a best influence on the output variable. In Fig. 3*b*–3*e*: *NB* indicates a low value; *NM*, a value closer to the average; *Z*, the average; *PM*, a value closer to a high value; *PB*, a high value. The output variable y is described by three terms in the form of the constants $t_{NB}^{y}=0$, $t_{Z}^{y}=50$, and $t_{PB}^{y}=100$ points (low, average, and high values, respectively). In view of the monograph [1], we assume that the partition-of-unity condition holds when constructing the MFs: for any crisp value of the variable x_i, the terms covering the corresponding segment of the crisp value axis have the grades of membership to the MFs adding up to 1. This condition can be written as $\sum_{k}\mu_{k}^{x_i}\left(x_i\right)\equiv 1\forall x_i\subset X$.

Due to the expressions (8) and (9), the normalization operators of the input variables are the following:

- under a positive correlation with the variable y,

$$x_{\text{norm}}^{\text{direct}}=\begin{cases} f_{\text{norm},1}\left(x_{\text{abs}}\right)=5\,x_{\text{abs}}\,|\,0\le x_{\text{abs}}\le 2,\\ f_{\text{norm},2}\left(x_{\text{abs}}\right)=7.5\,x_{\text{abs}}-5\,|\,2<x_{\text{abs}}\le 6,\\ f_{\text{norm},3}\left(x_{\text{abs}}\right)=10\,x_{\text{abs}}-20\,|\,6<x_{\text{abs}}\le 8,\\ f_{\text{norm},4}\left(x_{\text{abs}}\right)=30\,x_{\text{abs}}-180\,|\,8<x_{\text{abs}}\le 9,\\ f_{\text{norm},5}\left(x_{\text{abs}}\right)=10\,x_{\text{abs}}\,|\,9<x_{\text{abs}}\le 10; \end{cases} \quad (10)$$

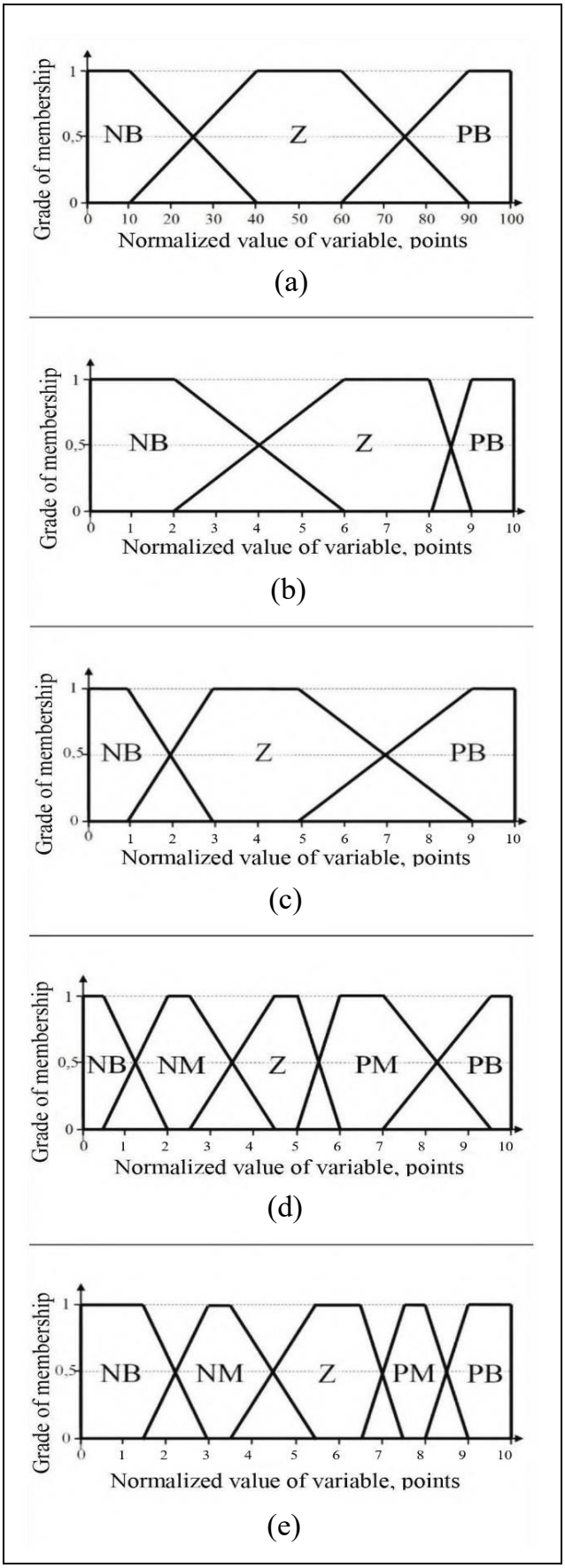


Fig. 3. Membership functions of input variables.

• under a negative correlation with the variable y,

$$x_{\text{norm}}^{\text{invers}} = \begin{cases} f_{\text{norm},1}(x_{\text{abs}}) = -10x_{\text{abs}} + 100 \,|\, 0 \le x_{\text{abs}} \le 1, \\ f_{\text{norm},2}(x_{\text{abs}}) = -15x_{\text{abs}} + 105 \,|\, 1 < x_{\text{abs}} \le 3, \\ f_{\text{norm},3}(x_{\text{abs}}) = -10x_{\text{abs}} + 90 \,|\, 3 < x_{\text{abs}} \le 5, \\ f_{\text{norm},4}(x_{\text{abs}}) = -7.5x_{\text{abs}} + 77.5 \,|\, 5 < x_{\text{abs}} \le 9, \\ f_{\text{norm},5}(x_{\text{abs}}) = -10x_{\text{abs}} + 100 \,|\, 9 < x_{\text{abs}} \le 10; \end{cases} \quad (11)$$

• under an optimum with respect to the variable y (variant 1),

$$x_{\text{norm}}^{\text{opt.var1}} = \begin{cases} f_{\text{norm},1\div2}(x_{\text{abs}}) = 20x_{\text{abs}} \,|\, 0 \le x_{\text{abs}} \le 2, \\ f_{\text{norm},3}(x_{\text{abs}}) = 40x_{\text{abs}} - 40 \,|\, 2 \le x_{\text{abs}} \le 2.5, \\ f_{\text{norm},4}(x_{\text{abs}}) = 15x_{\text{abs}} + 22.5 \,|\, 2.5 \le x_{\text{abs}} \le 4.5, \\ f_{\text{norm},5}(x_{\text{abs}}) = 40x_{\text{abs}} - 90 \,|\, 4.5 \le x_{\text{abs}} \le 4.75, \\ f_{\text{norm},6}(x_{\text{abs}}) = -40x_{\text{abs}} + 290 \,|\, 4.75 \le x_{\text{abs}} \le 5, \\ f_{\text{norm},7}(x_{\text{abs}}) = -30x_{\text{abs}} + 240 \,|\, 5 \le x_{\text{abs}} \le 6, \\ f_{\text{norm},8}(x_{\text{abs}}) = -20x_{\text{abs}} + 180 \,|\, 6 \le x_{\text{abs}} \le 7, \\ f_{\text{norm},9}(x_{\text{abs}}) = -12x_{\text{abs}} + 124 \,|\, 7 \le x_{\text{abs}} \le 9.5, \\ f_{\text{norm},10}(x_{\text{abs}}) = -20x_{\text{abs}} + 200 \,|\, 9.5 \le x_{\text{abs}} \le 10; \end{cases} \quad (12)$$

• under an optimum with respect to the variable y (variant 2):

$$x_{\text{norm}}^{\text{opt.var2}} = \begin{cases} f_{\text{norm},1}(x_{\text{abs}}) = \frac{20}{3}x_{\text{abs}} \,|\, 0 \le x_{\text{abs}} \le 1.5, \\ f_{\text{norm},2}(x_{\text{abs}}) = 20x_{\text{abs}} - 20 \,|\, 1.5 \le x_{\text{abs}} \le 3, \\ f_{\text{norm},3}(x_{\text{abs}}) = 40x_{\text{abs}} - 80 \,|\, 3 \le x_{\text{abs}} \le 3.5, \\ f_{\text{norm},4}(x_{\text{abs}}) = 15x_{\text{abs}} + 7.5 \,|\, 3.5 \le x_{\text{abs}} \le 5.5, \\ f_{\text{norm},5}(x_{\text{abs}}) = 20x_{\text{abs}} - 20 \,|\, 5.5 \le x_{\text{abs}} \le 6, \\ f_{\text{norm},6}(x_{\text{abs}}) = -20x_{\text{abs}} + 220 \,|\, 6 \le x_{\text{abs}} \le 6.5, \\ f_{\text{norm},7}(x_{\text{abs}}) = -30x_{\text{abs}} + 285 \,|\, 6.5 \le x_{\text{abs}} \le 7.5, \\ f_{\text{norm},8}(x_{\text{abs}}) = -40x_{\text{abs}} + 360 \,|\, 7.5 \le x_{\text{abs}} \le 8, \\ f_{\text{norm},9}(x_{\text{abs}}) = -30x_{\text{abs}} + 280 \,|\, 8 \le x_{\text{abs}} \le 9, \\ f_{\text{norm},10}(x_{\text{abs}}) = -10x_{\text{abs}} + 100 \,|\, 9 \le x_{\text{abs}} \le 10. \end{cases} \quad (13)$$

In all experiments, the FISs of type 1 had the same surface (Fig. 4*a*). The other FIS parameters were as follows:

– In experiment 1, the variables x_1 and x_2 in the FIS of type 1 were normalized using the operator (10). The variables x_1 and x_2 in the FIS of type 2 positively correlated with the variable y (Fig. 4*b*).

– In experiment 2, the variables x_1 and x_2 in the FIS of type 1 were normalized using the operator (11). The variables x_1 and x_2 in the FIS of type 2 negatively correlated with the variable y (Fig. 4*c*).

– In experiment 3, the variables x_1 and x_2 in the FIS of type 1 were normalized using the operators (12) and (13). The variables x_1 and x_2 in the FIS of type 2 had an optimum with respect to the variable y (Fig. 4*d*).

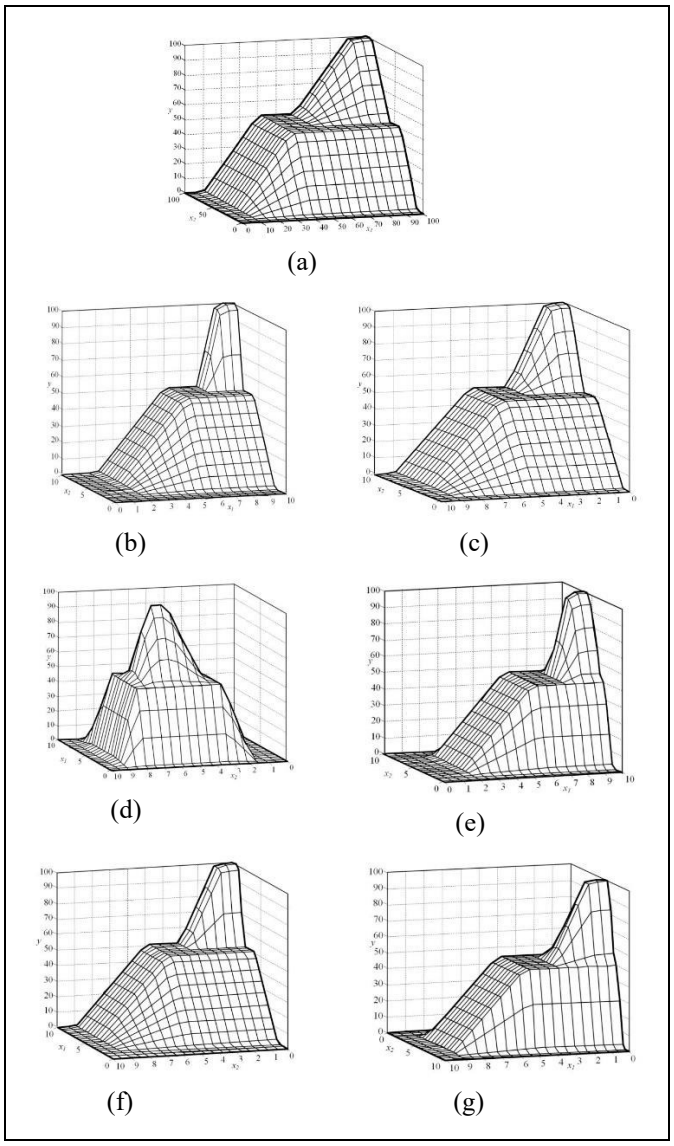


Fig. 4. Surfaces of FISs under comparison.

– In experiment 4, the variables x_1 and x_2 in the FIS of type 1 were normalized using the operators (10) and (13), respectively. The variable x_1 in the FIS of type 2 positively correlated with the variable y, whereas the variable x_2 in the FIS of type 2 had an optimum with respect to the variable y (Fig. 4*e*).

– In experiment 5, the variables x_1 and x_2 in the FIS of type 1 were normalized using the operators (10) and (11), respectively. The variables x_1 and x_2 in the FIS of type 2 positively and negatively correlated, respectively, with the variable y (Fig. 4*f*).

– In experiment 6, the variables x_1 and x_2 in the FIS of type 1 were normalized using the operators (11) and (13), respectively. The variable x_1 in the FIS of type 2 negatively correlated with the variable y, whereas the variable x_2 in the FIS of type 2 had an optimum with respect to the variable y (Fig. 4*g*).

During the experiments, the variables were supplied to the FIS input in the following values combinations in the range from 0 to 10 points:

– equal values of the variables with a step of 0.5 points (22 values of each variable in total),

– the values of the variable x_1 with a step of 0.5 points, and the fixed values of the variable x_2 starting from 0, increased by 0.5 points for each subsequent series of the experiment (462 values of each variable in total).

The error of the results was determined as the difference between the FISs of types 1 and 2:

$$\Delta_{\exp N,w} = \left| y^{\text{FIS}m1}_{\text{out},\exp N,w} - y^{\text{FIS}m2}_{\text{out},\exp N,w} \right|, \qquad (14)$$

where N denotes the experiment number; w is the number of the vector of values supplied to the FIS input during the experiment; $y^{\text{FIS}m1}_{\text{out},\exp N,w}$ and $y^{\text{FIS}m2}_{\text{out},\exp N,w}$ indicate the outputs of the FISs of types 1 and 2, respectively. In experiments 1–6, it was established that there are no errors between the output values of the FIS of type 1 with the proposed normalization method and the FIS of type 2 without normalization. For explaining the reasons, the features of the FIS operation were analyzed.

The FIS output is the accumulated result of activating the rules to construct the knowledge base. The result of each rule is determined by the grades of membership of the crisp values of the input variables to each term and the aggregation operators of their fuzzified values. In the case under consideration, the results of rule execution were processed without any changes, and the final FIS value was formed without any changes as well. The changes were introduced when preparing the variables (before supplying them to the FIS input).

Therefore, the FISs will operate without errors if the variables have the same values after the fuzzification procedure, regardless of using the proposed normalization method. This assertion was verified in a separate series of experiments. Below, we compare the values of the input variables after fuzzification for the MFs shown in Fig. 3*a* (the FISs with variables normalization) and MFs shown in Fig. 3*d* (the FISs with a variable having an optimum of its influence on the final result).

In Fig. 3*a*, the MFs are given by:

$$\mu^{3\text{a}}_{NB}(x) = \begin{cases} \mu^{3\text{a}}_{NB}(x) = 1 \mid 0 \le x < 10, \\ \mu^{3\text{a}}_{NB}(x) = -\dfrac{1}{30}x + \dfrac{4}{3} \mid 10 \le x < 40, \\ \mu^{3\text{a}}_{NB}(x) = 0 \mid x > 40, \end{cases} \qquad (15)$$

$$\mu^{3\text{a}}_{Z}(x) = \begin{cases} \mu^{3\text{a}}_{Z}(x) = 0 \mid 0 \le x < 10, \\ \mu^{3\text{a}}_{Z}(x) = \dfrac{1}{30}x - \dfrac{1}{3} \mid 10 \le x < 40, \\ \mu^{3\text{a}}_{Z}(x) = 1 \mid 40 \le x < 60, \\ \mu^{3a}_{Z}(x) = \dfrac{1}{30}x + 3 \mid 60 \le x < 90, \\ \mu^{3a}_{Z}(x) = 0 \mid x > 90, \end{cases} \qquad (16)$$

$$\mu^{3\text{a}}_{Z}(x) = \begin{cases} \mu^{3\text{a}}_{PB}(x) = 0 \mid x < 60, \\ \mu^{3\text{a}}_{PB}(x) = \dfrac{1}{30}x - 2 \mid 60 \le x \le 90, \\ \mu^{3\text{a}}_{PB}(x) = 0 \mid x > 90. \end{cases} \qquad (17)$$

In Fig. 3*d*, the MFs are given by:

$$\mu^{3\text{г}}_{NB}(x) = \begin{cases} \mu^{3\text{г}}_{NB}(x) = 1 \mid 0 \le x < 0{,}5, \\ \mu^{3\text{г}}_{NB}(x) = -\dfrac{2}{3}x + \dfrac{4}{3} \mid 0{,}5 \le x < 2, \\ \mu^{3\text{г}}_{NB}(x) = 0 \mid x > 2, \end{cases} \qquad (18)$$

$$\mu^{3\text{г}}_{NM}(x) = \begin{cases} \mu^{3\text{г}}_{NM}(x) = 0 \mid 0 \le x < 2, \\ \mu^{3\text{г}}_{NM}(x) = \dfrac{2}{3}x - \dfrac{1}{3} \mid 2 \le x < 6, \\ \mu^{3\text{г}}_{NM}(x) = 1 \mid 2 \le x < 2{,}5, \\ \mu^{3\text{г}}_{NM}(x) = -\dfrac{1}{2}x + 2\dfrac{1}{4} \mid 2{,}5 \le x < 4{,}5, \\ \mu^{3\text{г}}_{NM}(x) = 0 \mid x > 4{,}5, \end{cases} \qquad (19)$$

$$\mu^{3\text{г}}_{Z}(x) = \begin{cases} \mu^{3\text{г}}_{Z}(x) = 0 \mid x \le 2{,}5, \\ \mu^{3\text{г}}_{Z}(x) = \dfrac{1}{2}x - 1\dfrac{1}{4} \mid 2{,}5 \le x < 4{,}5, \\ \mu^{3\text{г}}_{Z}(x) = 1 \mid 4{,}5 \le x < 5, \\ \mu^{3\text{г}}_{Z}(x) = -x + 6 \mid 5 \le x \le 6, \\ \mu^{3\text{г}}_{Z}(x) = 0 \mid x > 6, \end{cases} \qquad (20)$$

$$\mu^{3\text{г}}_{PM}(x) = \begin{cases} \mu^{3\text{г}}_{PM}(x) = 0 \mid x < 5, \\ \mu^{3\text{г}}_{PM}(x) = x - 5 \mid 5 \le x < 6, \\ \mu^{3\text{г}}_{PM}(x) = 1 \mid 6 \le x < 7, \\ \mu^{3\text{г}}_{PM}(x) = -\dfrac{2}{5}x + 3\dfrac{4}{5} \mid 7 \le x < 9{,}5, \\ \mu^{3\text{г}}_{PM}(x) = 0 \mid x > 9{,}5, \end{cases} \qquad (21)$$

$$\mu^{3d}_{PB}(x) = \begin{cases} \mu^{3d}_{PB}(x) = 0 \mid x < 7, \\ \mu^{3d}_{PB}(x) = \dfrac{2}{5}x - 2\dfrac{4}{5} \mid 7 \le x \le 9.5, \\ \mu^{3d}_{PB}(x) = 1 \mid x > 9.5. \end{cases} \qquad (22)$$

Also, note the semantic equivalence of the following terms in Figs. 3*a* and 3*d*: $NB_{3a} \Leftrightarrow NB_{3d}$, $NB_{3a} \Leftrightarrow PB_{3d}$, $Z_{3a} \Leftrightarrow NM_{3d}$, $Z_{3a} \Leftrightarrow PM_{3d}$, and $PB_{3a} \Leftrightarrow Z_{3d}$. The values of the input variables are given in Table 2.

For clarity, the fuzzified values of the input variables normalized by the proposed method are highlighted in gray.

With the semantic equivalence of the terms, the errors using a formula similar to (14) were determined as follows. First, columns 3 and 7, as well as 4 and 6, were summed elementwise. Then:

– From the elementwise sum of columns 3 and 7, the corresponding elements of column 8 were subtracted.

– From the elementwise sum of columns 4 and 6, the corresponding elements of column 9 were subtracted.

– From the elements of column 5, the corresponding elements of column 10 were subtracted.

These operations yielded arrays with zero elements. The results of such experiments with the MFs shown in Figs. 3*b*, 3*c*, and 3*e* were similar. Under the experiment restrictions, such an accuracy was achieved because the interval limits on the normalization operator graph coincided with the points of the MF cores for the terms of the normalized input variables.

Due to the proposed modifications, six types of different FISs (Figs. 4*b*–4*g*) were replaced by one type of FIS (Fig. 4*a*). Experiment 3 showed a 64% reduction in the number of rules (9 rules in the FIS of type 1 vs. 25 rules in the FIS of type 2). Experiments 5 and 6 showed a 40% reduction in the number of rules (9 rules in the FIS of type 1 vs. 15 rules in the FIS of type 2). In experiment 3, both variables aggregated had one optimum of the influence on the value of the output variable; in experiments 5 and 6, only one variable had an optimum of the influence on the value of the output variable.

In addition, the time to form the rule bases was compared for the FISs of types 1 and 2 in experiments 2–6. Such a comparison was not carried out in experiment 1: the knowledge bases of the FISs of types 1 and

Table 2

Values for the terms of MFs in Figs. 3*a* and 3*d*

Input values		Fuzzified values of input variables:							
1	2	3	4	5	6	7	8	9	10
Without normalization	Normalized using operator (12)	(18) – $\mu_{NB}^{3d}(x)$	(19) – $\mu_{NM}^{3d}(x)$	(20) – $\mu_{Z}^{3d}(x)$	(21) – $\mu_{PM}^{3d}(x)$	(22) – $\mu_{PB}^{3d}(x)$	(15) – $\mu_{NB}^{3a}(x)$	(16) – $\mu_{Z}^{3a}(x)$	(17) – $\mu_{PB}^{3a}(x)$
0.5	10	1	0	0	0	0	1	0	0
1	20	0.667	0.333	0	0	0	0.667	0.333	0
1.5	30	0.333	0.667	0	0	0	0.333	0.667	0
2	40	0	1	0	0	0	0	1	0
2.5	60	0	1	0	0	0	0	1	0
3	67.5	0	0.75	0.25	0	0	0	0.75	0.25
3.5	75	0	0.5	0.5	0	0	0	0.5	0.5
4	82.5	0	0.25	0.75	0	0	0	0.25	0.75
4.5	90	0	0	1	0	0	0	0	1
5	90	0	0	1	0	0	0	0	1
5.5	75	0	0	0.5	0.5	0	0	0.5	0.5
6	60	0	0	0	1	0	0	1	0
6.5	50	0	0	0	1	0	0	1	0
7	40	0	0	0	1	0	0	1	0
7.5	34	0	0	0	0.8	0.2	0.2	0.8	0
8	28	0	0	0	0.6	0.4	0.4	0.6	0
8.5	22	0	0	0	0.4	0.6	0.6	0.4	0
9	16	0	0	0	0.2	0.8	0.8	0.2	0
9.5	10	0	0	0	0	1	1	0	0
10	0	0	0	0	0	1	1	0	0

2 contain the same rules. For each group of the FISs of types 1 and 2, five rule bases were formed. As a result, the rule base formation time when using the FIS of type 1 instead of the FIS of type 2 was:

- in experiments 2 and 5, by approximately 17% smaller, due to various types of correlations between the input variables and the output variable (no need for the expert to compare the semantic value of the term describing the input variable range and its influence on the final result);
- in experiments 4 and 6, by approximately 37% smaller, due to the number of rules in the knowledge base of the FIS of type 1 (9 rules) and the FIS of type 2 (15 rules);
- in experiment 3, by approximately 58% smaller, due to the number of rules in the knowledge base of the FIS of type 1 (9 rules) and the FIS of type 2 (25 rules).

Thus, the average reduction in the knowledge base formation time over the entire series of experiments 3–6 was about 35%.

3. AN EXAMPLE OF IMPLEMENTING THE PROPOSED MODIFICATIONS

As an example, consider a simplified FIS for ranking bank clients by creditworthiness. The decision on issuing a loan is based on three parameters:

- x_1, the borrower's monthly income, ranging from 15 to 100 thousand RUB. This variable positively correlates with the output variable: the higher the person's income is, the more likely he/she will get a loan;
- x_2, the borrower's share of monthly payments for serving the current loans. It is measured as a percentage of his/her earnings, ranging from 0 to 60%. The greater the person's share of current payments is, the less likely he/she will get a new loan;
- x_3, the borrower's age, ranging from 14 to 85 years. The best interval of this variable is between 35 to 45 years: at this age, a person has the highest "life" stability in terms of income and health.

The purpose of the experiment was to compare the results yielded by model 1 (with the proposed normalization method), model 2 (without normalization of the input variables), and model 3 (with the normalization method described in [2]: the values of the input variables were divided into intervals, and an interval on the normalized scale was assigned to each interval on the absolute axis). The models were FISs with three inputs (the variables x_1, x_2, and x_3) and one output (the variable y). The structure of models 1 and 3 is shown in Fig. 5*a*; the structure of model 2, in Fig. 5*b*.

In model 1, the variables x_1, x_2, and x_3 were described by the MFs shown in Fig. 3*a*. In model 2, the variables x_1, x_2, and x_3 were described by the MFs shown in Figs. 6*a*–6*c*.

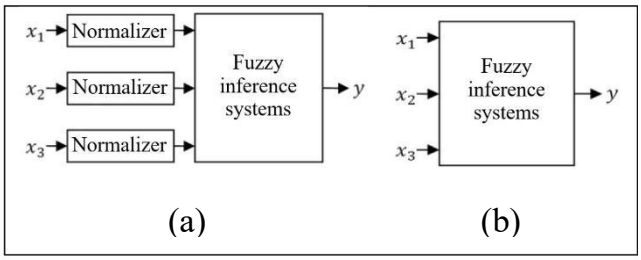


Fig. 5. The structure of studied models.

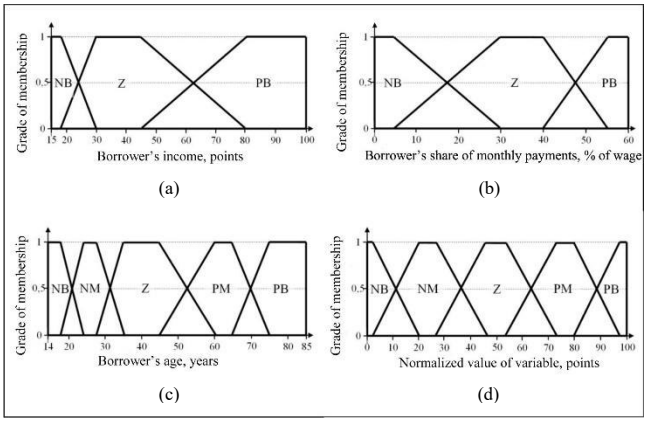


Fig. 6. Membership functions of input variables for studied models.

In model 3, the variables x_1 and x_2 were described by the MFs shown in Fig. 3*a*; the variable x_3, by the MFs shown in Fig. 6*d*. The semantic meanings of the terms in model 1 corresponded to those in Fig. 3*a*. The semantic meanings of the terms in models 2 and 3 corresponded to those in Fig. 3*b*–3*d*. The term set {*NB*, *NM*, *Z*, *PM*, *PB*}, where $NB \sim 0$, $NM \sim 25$, $Z \sim 50$, $PM \sim 75$, and $PB \sim 100$ points, was used to describe the output variable y. The term values of the output variable in all models were similar to those of the models of Section 2. The meaning of the term NM was "closer to the average"; PM, "closer to a high value."

The normalization operators for the input variables of model 1 were described by the following relations:

– for the variable x_1,

$$x_{1,\text{norm}} = \begin{cases} f_{\text{norm},1}(x_1) = (10/3)x_1 - 50 \,|\, 15 \le x_1 \le 18, \\ f_{\text{norm},2}(x_1) = 2.5x_1 - 35 \,|\, 18 < x_1 \le 30, \\ f_{\text{norm},3}(x_1) = (4/3)x_1 \,|\, 30 < x_1 \le 45, \\ f_{\text{norm},4}(x_1) = (6/7)x_1 + (150/7) \,|\, 45 < x_1 \le 80, \\ f_{\text{norm},5}(x_1) = 0.5x_1 + 50 \,|\, 80 < x_1 \le 100; \end{cases}$$

– for the variable x_2,

$$x_{2,\text{norm}} = \begin{cases} f_{\text{norm},1}(x_2) = -2x_2 + 100 \,|\, 0 \le x_2 \le 5, \\ f_{\text{norm},2}(x_2) = -1.2x_2 + 96 \,|\, 5 < x_2 \le 30, \\ f_{\text{norm},3,4,5}(x_2) = -2x_2 + 120 \,|\, 30 < x_2 \le 60; \end{cases}$$

– for the variable x_3,

$$x_{3,\text{norm}} = \begin{cases} f_{\text{norm},1}(x_3) = 2.5x_3 - 35 \,|\, 14 \le x_3 \le 18, \\ f_{\text{norm},\,2,3}(x_3) = 5x_3 - 80 \,|\, 18 \le x_3 \le 28, \\ f_{\text{norm},4}(x_3) = \dfrac{30}{7}x_3 - 60 \,|\, 28 \le x_3 \le 35, \\ f_{\text{norm},5}(x_3) = 2x_3 + 20 \,|\, 35 \le x_3 \le 40, \\ f_{\text{norm},6,7}(x_3) = -2x_3 + 180 \,|\, 40 \le x_3 \le 60, \\ f_{\text{norm},8}(x_3) = -4x_3 + 300 \,|\, 60 \le x_3 \le 65, \\ f_{\text{norm},9}(x_3) = -3x_3 + 235 \,|\, 65 \le x_3 \le 75, \\ f_{\text{norm},10}(x_3) = -1x_3 + 85 \,|\, 75 \le x_3 \le 85. \end{cases}$$

The normalization operators for the input variables of model 3 were implemented considering the method described in [2]. Table 3 compares the intervals of the parameter values in absolute and normalized units.

Table 3

Variable normalization for model 3

Variable and its measurement unit	Absolute-value intervals	Normalized-value intervals
x_1 (income, thousand RUB)	$15 \le x_1 \le 18$	[0, 10]
	$18 < x_1 \le 30$	[10, 40]
	$30 < x_1 \le 45$	[40, 60]
	$45 < x_1 \le 80$	[40, 90]
	$80 < x_1 \le 100$	[90, 100]
x_2 (the share of monthly payments on current loans, % of income)	$0 \le x_2 \le 5$	[0, 10]
	$5 < x_2 \le 30$	[10, 40]
	$30 < x_2 \le 40$	[40, 60]
	$40 < x_2 \le 55$	[40, 90]
	$55 < x_2 \le 60$	[90, 100]
x_3 (age, years)	$14 \le x_3 \le 18$	[0, 2.5]
	$18 < x_3 \le 24$	[2.5, 20]
	$24 < x_3 \le 28$	[20, 27]
	$28 < x_3 \le 35$	[27, 46]
	$35 < x_3 \le 45$	[46, 54]
	$45 < x_3 \le 60$	[54, 73]
	$60 < x_3 \le 65$	[73, 80]
	$65 < x_3 \le 75$	[80, 97.5]
	$75 < x_3 \le 85$	[97.5, 100]

Various value combinations of the variables x_1, x_2, and x_3 were supplied to the input of the studied models: 100 different combinations of the absolute values in total. The values were random numbers between the maximum and minimum values of the corresponding parameter. The results yielded by models 1, 2, and 3 are shown in Figs. 7*a*, 7*b*, and 7*c*, respectively.

According to the experiment results, the total absolute error between models 1 and 2 is approximately 0 points;

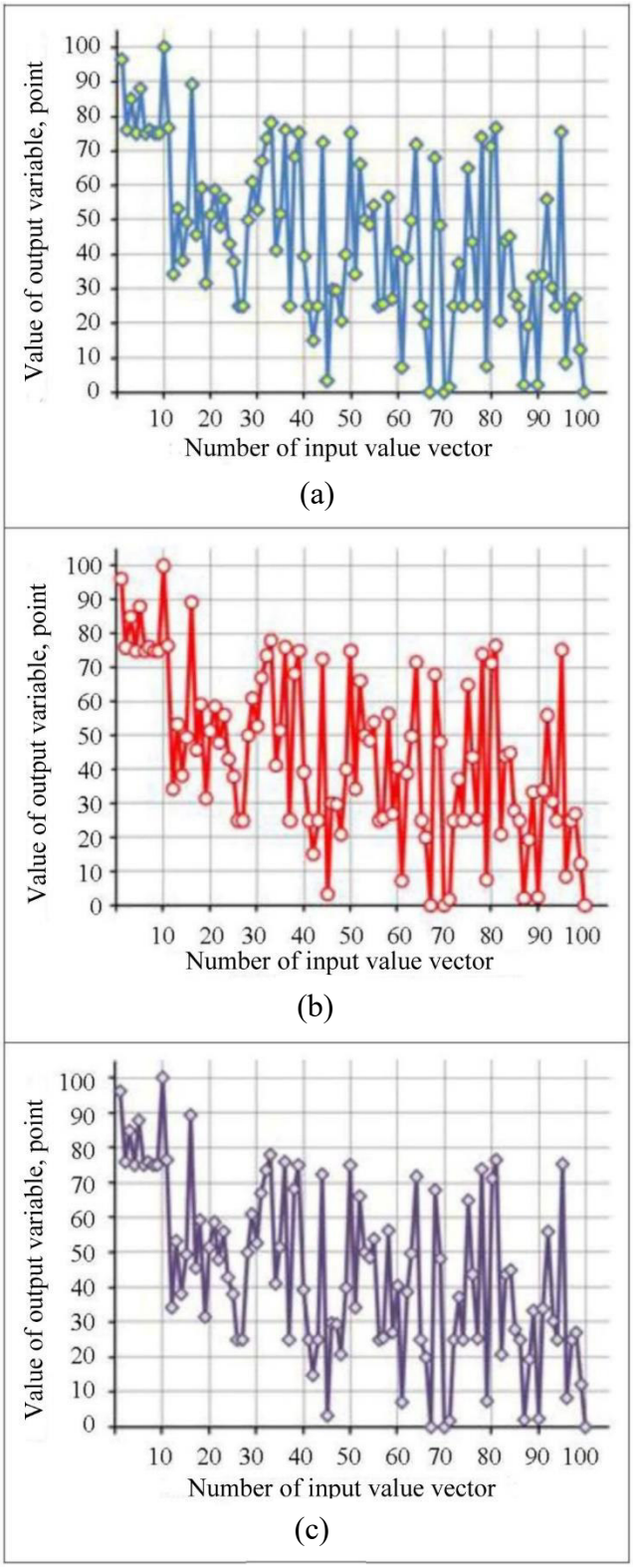


Fig. 7. Client's creditworthiness: experiment results.

between models 2 and 3, approximately 0 points. "Approximately" means that the maximum error value in the experiments did not exceed 10^{-13}. Thus, the models yield almost the same output values. Such an accuracy was achieved due to normalizing the variables of models 1 and 2 similar to the experiment of subsection 2.2. (The results of this experiment are shown in Figs. 4*a*–4*g*.) The experiment carried out confirms the reliability of the proposed normalization method in the formation of model 1. The knowledge bases of models 2 and 3 contain 45 rules each; the knowledge base of model 1, 27 rules. Consequently, with the proposed modifications, model 1 has the same accuracy of calculations under a smaller (by 40%) number of rules in the knowledge base.

Similar to the previous experiment (see Section 2), the knowledge base formation time was compared for the FISs in models 1 and 2. (Model 3 was not considered due to the same rules as in model 2.) By analogy, five knowledge bases were formed for the FISs of models 1 and 2. As a result of the comparison, it was found that the knowledge base formation time of model 1 is by approximately 47% smaller than that of model 2. This difference can be explained as follows. The knowledge base of model 2 contains more rules compared to model 1 (45 vs. 27). Also, all input variables in model 2 have different types of correlation with the output variable.

4. DISCUSSION OF THE RESULTS

The proposed approach for transforming the values of input variables can be implemented using software methods: it reduces to constructing an array of straight-line equations with two given points. According to the experimental evidence of this paper, if the variable under normalization has an optimum of the influence on the final result, the number of production rules in the FIS knowledge base will be reduced. As is known, the formation of production rules consumes much effort, often manually performed by an expert. Thus, the proposed modifications allow decreasing the labor costs of expert groups when forming knowledge bases. Here, an additional difficulty for experts is caused by different properties of the input variables due to the variety of ranges, measurement units, and influences on the value of the output variable. Under such a variety, an expert has to remember the specifics of each variable, analyzing the input variables more carefully during rules formation. According to the paper [19], the increased concentration of person's attention leads to rapid fatiguability, negatively affecting the number of mistakes allowed. Eliminating mistakes is time-consuming. With the proposed modifications, the normalized variables have a single range and the same correlation with the output variable. Such homogeneous properties of the input variables simplify the process of forming production rules.

Despite the introduction of additional mathematical operations, the experiments have not revealed a significant decrease in the computing performance of the hardware means of the FISs under consideration. This is due to the large computing power reserve of modern computers. The proposed modifications have a wide application area and are intended to implement expert systems for the integral assessment of a complex object. Therefore, the increasing computational cost can be considered insignificant compared to the resulting simplification of the knowledge base formation process. As is known, this process is mainly performed directly by experts. In the experiments of Sections 2 and 3, less time was spent on FIS knowledge base formation using the proposed modifications (on average, by approximately 37% and 47%, respectively) than on FIS knowledge base formation without normalization.

As follows from the monograph [1], the curse of dimensionality is a problem of FIS construction. More specifically, the number of rules in the knowledge base strongly depends on the number of variables n_{var} and the number of terms n_{term} describing each variable. For example, if the same number of terms is used to describe all input variables, then the number of MISO rules is given by $n_{rule} = {n_{term}}^{n_{var}}$. The negative influence of the curse of dimensionality can be decreased by developing modifications that will reduce the number of rules in the knowledge base while maintaining the accuracy of FISs. According to the experimental evidence of this paper, the proposed modifications allow reducing the number of rules in the knowledge base depending on the experiment conditions: by 40% in the experiment on the borrower's creditworthiness (Section 3) and in experiments 5 and 6 (Section 2); by 64% in experiment 3 (Section 2).

CONCLUSIONS

This paper has proposed a method for normalizing the input variables of fuzzy inference systems. The method transforms the absolute values of input variables to a single range of values in normalized units.

Note that the minimum value of the normalized variable has the worst influence on the output parameter, whereas the maximum value of the normalized variable has the best influence on it. The method divides the variable's range into a sequence of intervals. Then a pattern is formed for each interval to transform the absolute values of the parameter to the normalized ones. The normalization operator on a given interval

is implemented by constructing a straight line with two given points. According to the modeling results, under the specified restrictions, the FIS models with the proposed normalization method are adequate to the FIS models without normalization of the input variables. The proposed normalization method allows reducing the number of rules in the knowledge base depending on the experiment conditions: by 40% in the experiment on the borrower's creditworthiness (Section 3) and in experiments 5 and 6 (Section 2); by 64% in experiment 3 (Section 2). A common property of fuzzy inference systems in these experiments has been the presence of input variables with an optimum of the influence on the value of the output variable. In the experiments presented above, less time has been spent on FIS knowledge base formation using the proposed modifications (on average, by approximately 37% and 47%, respectively) than on FIS knowledge base formation without normalization.

The proposed modifications open up opportunities for developing information processing methods for decision support systems of various purposes.

REFERENCES

1. Piegat, A., *Fuzzy Modeling and Control*, Heidelberg: Physica-Verlag, 2001.
2. Lee, K.H., *First Course on Fuzzy Theory and Applications*, Springer Science & Business Media, 2004.
3. Amindousta, A., Ahmeda, S., Saghafiniab, A., and Bahreininejada, A., Sustainable Supplier Selection: A Ranking Model Based on Fuzzy Inference System, *Applied Soft Computing*, 2012, no. 12, pp. 1668–1677.
4. Alavi, N., Date Grading Using Rule-Based Fuzzy Inference System, *Journal of Agricultural Technology*, 2012, no. 8(4), pp. 1243–1254.
5. Cavallar, F., A Takagi–Sugeno Fuzzy Inference System for Developing a Sustainability Index of Biomass, *Sustainability*, 2015, no. 7(9), pp. 12359–12371.
6. Latinovic, M., Dragovic, I., Arsic, V.B., and Petrovic, B., A Fuzzy Inference System for Credit Scoring Using Boolean Consistent Fuzzy Logic, *International Journal of Computational Intelligence Systems*, 2018, vol. 11, iss. 1, pp. 414–427.
7. Novacovich, B., Vranjes, B., and Novacovich, D., An Optimal Adaptation Algorithm for Fuzzy Logic Control Systems, *Proceedings of 6th IEEE Mediterranean Conference on Theory and Practice of Control and Systems*, Alterhero, 1998, pp. 629–634.
8. Grassian, D., Bahatem, M., Scott, T., and Olsen, D., Application of a Fuzzy Expert System to Analyze and Anticipate ESP Failure Modes, *Abu Dhabi International Petroleum Exhibition & Conference*, Abu Dhabi, 2017, pp. 2–10.
9. Bermudez, F., Carvajal, G.A., Moricca, G., et al., Fuzzy Logic Application to Monitor and Predict Unexpected Behavior in Electric Submersible Pumps (Part of the KwIDF Project), *SPE Intelligent Energy Conference & Exhibition*, Utrecht, 2014, pp. 1–13.
10. Lee, E., Choi, C., and Kim, P., Intelligent Handover Scheme for Drone Using Fuzzy Inference Systems, *IEEE Access*, 2017, vol. 5, pp. 13712–13719.
11. Zainol-Abidin, S.N., Jaaman, S.H., Ismail, M., and Abu-Bakar, A.S., Clustering Stock Performance Considering Investor Preferences Using a Fuzzy Inference System, *Symmetry*, 2020, vol. 12, iss. 7, pp. 1–15.
12. Özger, M., Comparison of Fuzzy Inference Systems for Stream Flow Prediction, *Hydrological Sciences Journal*, 2009, vol. 54, pp. 261–273.
13. Sonmez, A.Y., Kale, S., Ozdemir, R.C., and Kadak, A.E., An Adaptive Neuro-Fuzzy Inference System (ANFIS) to Predict of Cadmium (Cd) Concentrations in the Filyos River, Turkey, *Turkish Journal of Fisheries and Aquatic Sciences*, 2018, vol. 18, pp. 1333–1343.
14. Shtovba, S.D., *Proektirovanie nechetkikh sistem sredstvami MATLAB* (Designing Fuzzy Systems in MATLAB), Moscow: Goryachaya Liniya-Telekom, 2007. (In Russian.)
15. Shmeleva, A.G., Kalenyuk, I.V., Obydennova, S.Yu., et al., A Software Model aor Assessing Customers' Credit Worthiness Using Artificial Intelligence Algorithms, *Trudy NGTU im. R.E. Alekseeva*, 2020, no. 3 (130), pp.72–79. (In Russian.)
16. Shakirov, V. and Pankrat'ev, P., Decision Making Support at the Pre-Feasibility Study Stage Based on Two Level Multi-Attribute Analysis, *Journal of Applied Informatics*, 2013, no. 6 (48), pp. 111–121. (In Russian.)
17. Nefedov, A.S. and Shakirov, V.A., Multi-criteria Assessment of Alternatives Based on the TOPSIS Method in the Conditions of Uncertainty of the Preferences of the Decision Maker, *Information Technologies. Problems and Solutions*, 2019, no. 3(8), pp. 25–32. (In Russian.)
18. Gvozdik, M.I., Abdulaliev, F.A., and Shilov, A.G., The Risk Assessment Model in Fuzzy Environment Using Logical Inference on Fuzzy Sets of the First Order, *Vestnik Sankt-Peterburgskogo Universiteta GPS MCHS Rossii*, 2017, no. 2, pp. 107–120. (In Russian.)
19. Akimova, G.P., Soloviev, A.V., and Pashkina, E.V. Methodological Approach to Determining the Influence of the Human Factor on the Performance of Information Systems, *Proceeding of the Institute for Systems Analysis of the Russian Academy of Sciences*, 2007, vol. 29, pp.102–112. (In Russian.)

This paper was recommended for publication by V.Yu. Stolbov, a member of the Editorial Board.

Received March 15, 2021, and revised June 7, 2021.
Accepted June 22, 2021.

Author information

Sorokin, Aleksandr Aleksandrovich. Cand. Sci. (Eng.), Astrakhan State Technical University, Astrakhan, Russia, ✉ alsorokin.astu@mail.ru

Cite this article

Sorokin, A.A., Using Piecewise Functions to Normalize Input Variables of Fuzzy Inference Systems. *Control Sciences* **4,** 54–66 (2021). http://doi.org/10.25728/cs.2021.4.6

Original Russian Text © Sorokin, A.A., 2021, published in *Problemy Upravleniya*, 2021, no. 4, pp. 66–81.

DOI: http://doi.org/10.25728/cs.2021.4.7

NON-BLOCKING FAULT-TOLERANT TWO-STAGE DUAL PHOTON SWITCHES

E.A. Barabanova, K.A. Vytovtov, and V.S. Podlazov

Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia

✉ elizavetaalexb@yandex.ru, ✉ vytovtov_konstan@mail.ru, ✉ podlazov@ipu.ru

Abstract. This paper further develops the theory of constructing a fundamentally new class of system area networks – non-blocking dual photon networks with static self-routing. These networks have scalability, high speed inherent in photon systems, and complexity comparable to a full switch. The use of an extended scheme basis (dual photon switches and separate photon multiplexers and demultiplexers) allows balancing the scalability-speed and complexity-speed ratios. This paper proposes a method for constructing a two-stage fault-tolerant dual network with the indicated properties based on networks with the quasi-complete graph and quasi-complete digraph topologies and the invariant extension method with internal parallelization.

Keywords: photon switch, dual switch, photon multiplexers and demultiplexers, multi-stage switch, conflict-free self-routing, non-blocking switch, static self-routing, quasi-complete digraph, quasi-complete graph, switching properties, direct channels, scalability, speed and fault tolerance.

INTRODUCTION

The papers [1–4] proposed a technique for constructing non-blocking photon switches with static self-routing for optical supercomputer systems and communication systems.

Below, we formulate and solve the problem of constructing non-blocking self-routing photon switches with $(\sigma - 1)$-channel fault tolerance. It is provided by replacing the dual quasi-complete digraph topology [1–3] with the dual quasi-complete graph one, in which the total number of channels can be exchanged for the number of different channels between users. This problem is solved in the same way: using the method of internal parallelization developed earlier in [4].

Non-blocking networks with static self-routing are a fundamentally new class of system area networks that achieve these properties by using a new type of non-blocking dual switches and the invariant extension method based on networks with the quasi-complete digraph and graph topologies [4, 5].

This assertion rests on the fact that no such networks with high scalability and acceptable complexity have been constructed so far. Modern literature [6–13] widely describes system area networks with the fat tree structure (particularly reconfigurable Clos networks), the generalized hypercube structure, the multidimensional torus structure, and system area networks with a hierarchy of complete graphs or digraphs.

Fat-tree networks are reconfigurable networks [6, 7] with conflict-free transmission only according to predetermined schedules for specific packet permutations. In the case of arbitrary permutations, these networks turn out to be blocking; permutations in them are implemented in several jumps between network nodes. Networks with the generalized hypercube structure are not even reconfigurable [8, 9]. They can be made such by increasing the number of channels in some dimensions. For arbitrary permutations, networks with the multidimensional torus structure cannot transmit packets over direct channels [10, 11]. They implement permutations in several jumps between network nodes. Networks with a hierarchy of complete graphs or digraphs have similar properties [12, 13].

In the papers [1–4], the method of constructing non-blocking photon switches with static self-routing was based on four fundamental principles:

1. Applying a four-channel switch of a new structure, which is dual by the conflict resolution approach. It combines the bus method (separation of conflicting signals to different cycles in one channel) and the switch method (separation of conflicting signals to different channels).
2. Assuming the parallel transmission of signaling and control information for switches at different frequencies for each data bit. This assumption eliminates the problem of synchronizing signals from different channels.
3. Cascading switches so that the *I*th channel of the *J*th switch on one stage is connected to the *J*th channel of the *I*th switch on the next stage. With such exchange links, the previous and next stages must include the same number of switches, each having the same number of channels. This method allows constructing multichannel switches with a small number of stages.
4. Balancing the speed and complexity of a multistage switch using the invariant extension method of system area networks [5], which preserves the non-blocking property and speed of the switch with an increase in the number of its channels. This method involves an extended scheme (circuit) basis consisting of both $p\times p$ switches for p channels and pairs of $1\times p$ multiplexers Mp and $p\times 1$ demultiplexers Dp, where $p \geq 2$.

In Section 1, the works on photonics close to the problem under study are considered; see [14–23]. Section 2 describes a technique for constructing non-blocking photon switches, used by the authors in the previous works [1–4]. Section 3 presents a switch with the dual quasi-complete graph topology as the basis for constructing non-blocking self-routing switches with channel fault tolerance. In Section 4, two-stage non-blocking self-routing switches with one-channel and two-channel fault tolerance are constructed; in addition, their characteristics – speed and complexity––are estimated. In the Conclusions, the results are summarized, and their possible development and generalization for constructing four-stage and eight-stage non-blocking fault-tolerant switches of high scalability are discussed.

1. RELEVANT RESEARCH INTO PHOTONICS: A SURVEY

Currently, a promising direction in the development of high-performance supercomputers is using photonic technologies [14, 15]. Much attention is paid to the design of photon switches that can significantly accelerate information transmission and processing [16–23]. In the paper [16], photon switches were constructed using such well-known topologies as Clos, Benes, the matrix switch, Dragonfly, etc. A disadvantage of these circuits is external electronic control: on the one hand, it allows for non-blocking switching; on the other hand, it restricts the speed of optical circuits on average to several tens of nanoseconds [17, 18]. For example, an externally controlled matrix switch with dimensions 240 × 240 was presented in [23] based on the MEMS technology. Its speed is 400 ns. Another problem in implementing optical switches is the need to use optical buffer devices [18]. In [22], a method was proposed for transmitting and processing information, which reduces the size of the optical buffer by 4.2 times and increases the speed by 2.6 times. In [18], a new principle of optical switching—segment switching – was proposed. It differs from channel switching: when establishing a connection, not the entire route from the source to the sink is reserved in advance, but its smaller segments. This approach improves network performance and reduces the number of conflicts.

In [14–23], individual components of photon networks were presented, but without any method for combining them into networks with specified properties. On the contrary, in [1–4], a technique for constructing non-blocking photon switches with static self-routing was described.

2. A METHOD FOR CONSTRUCTING NON-BLOCKING PHOTON SWITCHES

In the papers [1–4], one of the circuits of a one-chip dual 4×4 switch is a two-stage circuit of four demultiplexers and four multiplexers with feedback links through the delay lines. The switch stages are interconnected by exchange links. If the complexity of multiplexers M4 and demultiplexers D4 is measured by the number of switching points (equal to 4), then the switching complexity of the switch is $S_1 = 32$. The combination of two control frequencies uniquely determines the demultiplexer's mode in which the information signal can be directed to one of the four outputs.

The signals from the demultiplexer's outputs are supplied to the multiplexer's inputs. One of them is passed to the output, and the rest return to their delay lines DLδ with a duration of δ cycles (signals). The switch implements dynamic signal delay using the feedback links through DLδ, which requires a higher period to transmit the signals.

The dual switch SS4 provides non-blocking with static self-routing under an appropriate length δ of the delay line. The value δ depends on the number of the stage in which the switch SS4 is used.

For the first stage, δ = 1. Let four signals of duration T_0 be simultaneously supplied to the inputs of the switch SS4, all received in one cycle. With dynamic signal delay at the switch outputs, there are four possible options of signal placement: one at each output, two signals in a row at two outputs, one and three signals in a row at two outputs, and four signals in a row at one output. As a result, the switch SS4 will be non-blocking for any input traffic when the period $\underline{T}_1$ of information signals is four cycles. Here, the underline denotes the values obtained in [1–3].

Therefore, the non-blocking self-routing switch SS4 has the following performance characteristics: the signal period $\underline{T}_1 = 4 = \underline{N}_1$ cycles, the number of channels $\underline{N}_1 = 4$, and the switching complexity $\underline{S}_1 = 32 = \underline{N}_1^{5/2}$.

The papers [1–3] considered the two-stage 16×16 switch $S_2$16 with exchange links, consisting of four switches SS4 on each stage. The first stage includes DL1, and the second stage includes DL0 (no delay lines). For an arbitrary permutation of packets, $S_2$16 turned out to be a non-blocking self-routing switch with the following performance characteristics: the number of channels $\underline{N}_2 = 16$, the signal period $\underline{T}_2 = 4 = \underline{N}_2^{1/2}$ cycles, and the switching complexity $\underline{S}_2 = 2{\cdot}4{\cdot}32 = 256 = \underline{N}_2^2$.

The papers [1–3] considered the four-stage 256×256 switch $S_4$256 with exchange links, consisting of 16 switches $S_2$16 on each stage. It consists of four stages of switches SS4. The first stage includes DL1, the second stage includes DL4, the third stage includes DL15, and the fourth stage includes DL0 (no delay lines). For an arbitrary permutation of packets, $S_4$256 turned out to be a non-blocking self-routing switch with the following performance characteristics: the signal period $\underline{T}_4 = 49 \approx 3\underline{N}_4^{1/2}$ cycles, the number of channels $\underline{N}_4 = 256$, and the switching complexity $\underline{S}_4 = 2{\cdot}16{\cdot}256 = 8192 = \underline{N}_4^{1.625}$. Note the large digit period (low speed) and small complexity of this switch.

In the papers [1–3], the "speed–complexity" ratio was balanced using the invariant extension of small-period switches. In particular, the switch $S_2$16 was extended through external multiplexers M4 and demultiplexers D4. This method of increasing the number of channels can be called the external parallelization of a non-blocking network. As a result, the non-blocking self-routing switch $S_3$64 was constructed, consisting of 16 switches $S_2$16 and 64 demultiplexers D4 and multiplexers M4. This switch has the following performance characteristics: the number of channels $\underline{N}_3 = 64$, the signal period $\underline{T}_3 = 4 = N_3^{1/3}$ cycles, and the switching complexity $\underline{S}_3 = 16{\cdot}256+4{\cdot}128 = 4\,608 = \underline{N}_3^{2.028}$.

In the paper [4], a non-blocking switch with the quasi-complete digraph topology was used not only to extend two-stage switches but immediately to construct them. The resulting switches have a higher speed (a smaller signal period) and more channels than those proposed in [1–3].

A switch with the quasi-complete digraph topology is constructed from dual $p \times p$ switches (DSp) together with $1 \times p$ demultiplexers (Dp) and $p \times 1$ multiplexers (Mp). The non-blocking switch SFN_1 with $N_1 = p^2$ channels consists of N_1 switches DSp, demultiplexers Dp, and multiplexers Mp. Its circuits for $p = 2$ and $p = 4$ are shown in Fig. 1 and 2, respectively. The values obtained below will be given without the underline.

The switch SFN_1 has the switching complexity $S_1 = 2pN_1 + 2p^2N_1 = 2p^3(p+1)$ and the channel complexity $L_1 = 2pN_1 = 2p^3$ (the number of channels without internal channels DSp). In Table 1, their values are expressed by an exponential relationship through the number of channels. In what follows, the complexity specified in this way will be called the exponential complexity.

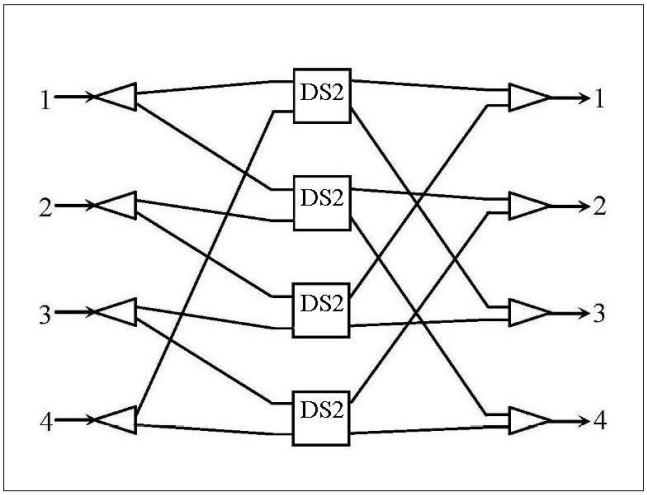


Fig. 1. Dual switch SF4 with quasi-complete graph topology. Boxes correspond to switches DS2, and triangles to demultiplexers D2 and multiplexers M2.

Table 1

Performance characteristics of switch SFN_1

p	2	4	6	8
N_1	4	16	36	64
S_1	$N_1^{2.79}$	$N_1^{2.33}$	$N_1^{2.24}$	$N_1^{2.19}$
L_1	N_1^2	$N_1^{1.75}$	$N_1^{1.69}$	$N_1^{1.67}$

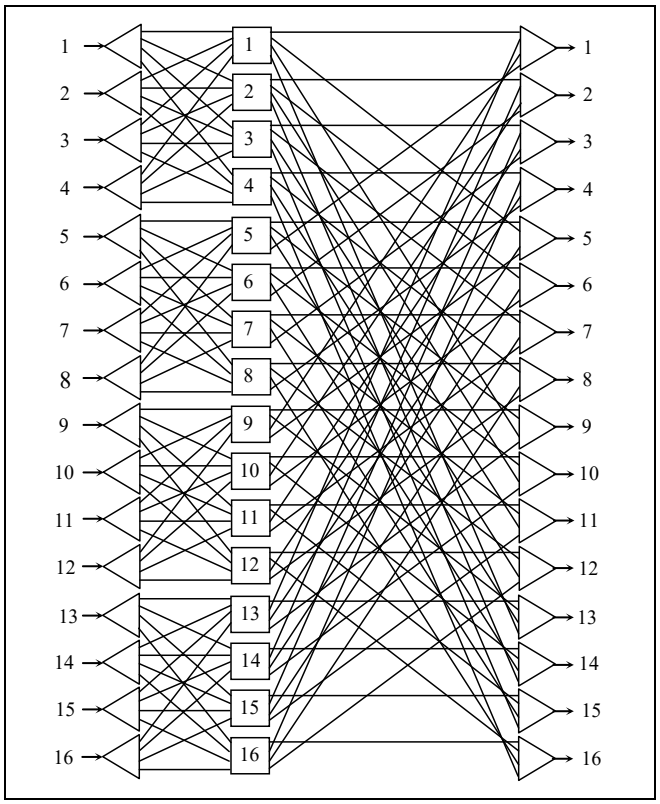


Fig. 2. Dual switch SF16 with quasi-complete digraph topology. Boxes correspond to switches DS4, and triangles to multiplexers M4 and demultiplexers D4.

From non-blocking dual switches SF4, it is possible to construct a two-stage network $N_2$16 with exchange links [5]. It consists of two stages with four SF4 on each stage. Unfortunately, the network $N_2$16 is a blocking network: implementing an arbitrary permutation in $N_2$16 is not reduced to implementing permutations on switches SF4 of the first and second stages. Signal conflicts can occur on a stage of multiplexers M2; see the gray segment in Fig. 5 of [5]. Signals with the same cycles can conflict at different multiplexer inputs.

The invariant internal parallelization method [4] transforms the network $N_2$16 into the non-blocking self-routing switch $S_2$16 with the signal period $T_2 = 2$ cycles.

For any $p > 2$, the same approach yields the non-blocking self-routing switch S_2N_2 with $N_2 = p^4$ channels and the digit period $T_2 = p$. The switching complexity of the switch S_2N_2 is given by the recursive formula $S_2 = N_1S_1 + pN_1S_1$; the channel complexity, by the recursive formula $L_2 = N_1L_1 + pN_1L_1$. The performance characteristics of the switch S_2N_2 are presented in Table 2.

Note that the switching and channel complexity of the switch S_2N_2 is significantly less than that of the switch SFN_1 (see Table 1). In other words, internal parallelization opens up the possibility of reducing the exponential complexity of a two-stage switch compared to the original switch SFN_1.

In the paper [4], the invariant extension method with external parallelization of non-blocking switches was also used through additional compound multiplexers Md and demultiplexers Dd with $d = p^{\alpha}$ ($1 \le \alpha \le 4$). It yielded a set of non-blocking self-routing switches with the highest scalability.

Well, the papers [1–4] considered various approaches to constructing non-blocking photon switches of high scalability. However, the fault tolerance of such networks was not touched upon. The photon switches [1–4] have only one physical path of information transmission between any "input – output" pair. In view of this feature, the development of new switching circuits with many such channels will increase their fault tolerance and is a topical problem.

In this paper, we construct non-blocking self-routing photon switches with (σ – 1)-channel fault tolerance. It is provided by replacing the dual quasi-complete digraph topology with the dual quasi-complete graph one, in which the total number of channels can be exchanged for the number of different channels between users. This problem is solved using the method of internal parallelization.

3. A NON-BLOCKING SELF-ROUTING SWITCH WITH THE QUASI-COMPLETE GRAPH TOPOLOGY

Dual switches DSp, together with demultiplexers Dp and multiplexers Mp, can be used to construct a non-blocking self-routing switch with the quasi-complete graph topology for N_1 channels – the dual switch SQG(N_1, p, σ) – in which $N_1 = p(p - 1)/\sigma + 1$ and σ specifies the number of duplex channels between any two users through different switches DSp.

The switch SQG(N, p, σ) is isomorphic to such a mathematical object as an incomplete balanced sym-

Table 2

Performance characteristics of switches S_2N_2

p	$N_2 = p^4$	$T_2 = p$	S_2	L_2
2	16	2	$576 = N_2^{2.29}$	$224 = N_2^{1.95}$
3	81	3	$7776 = N_2^{2.04}$	$2187 = N_2^{1.75}$
4	256	4	$51\,200 = N_2^{1.96}$	$11\,264 = N_2^{1.68}$
5	625	5	$225\,000 = N_2^{1.91}$	$40\,625 = N_2^{1.65}$
6	1296	6	$762\,048 = N_2^{1.89}$	$116\,640 = N_2^{1.63}$
7	2401	7	$2\,151\,296 = N_2^{1.87}$	$285\,719 = N_2^{1.61}$
8	4096	8	$5\,308\,416 = N_2^{1.86}$	$622\,592 = N_2^{1.60}$

metric block design $B(N, p, \sigma)$ [24]. It contains N blocks and N elements arranged so that each block contains exactly p different elements, each element appears in exactly p different blocks, and each pair of elements appears in exactly σ blocks. The block design $B(N, p, \sigma)$ sets the maximum value of N for given values of p and σ.

In the switch representation, blocks are interpreted as switches DSp; elements, as users of degree p (with p duplex ports); the entry of an element into a block, as a connection of each switch DSp to each user through a duplex channel. In this case, a single-port user is connected to different switches DSp through a pair of demultiplexers Dp and multiplexers Mp. Figure 3 shows the circuit of the switch SQG(7, 4, 2) with duplex channels and a combined pair of demultiplexers Dp and multiplexers Mp in one duplex channel splitter (hub).

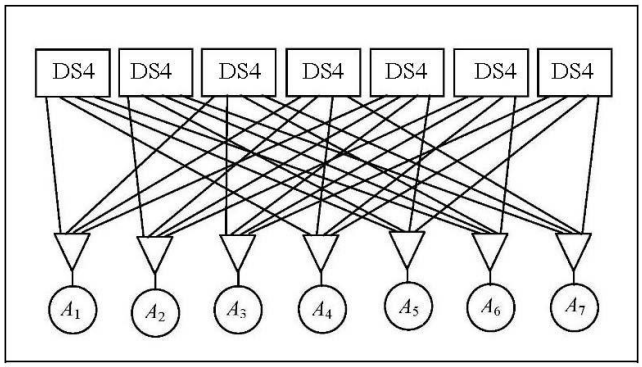


Fig. 3. Non-blocking network based on switch SQG(7, 4, 2) with two different channels between any users.

To construct a multichannel non-blocking network, we use the switch SQG(N, p, σ) with dividing each duplex channel into two simplex ones: the input channel from the demultiplexers Dp and the output channel from the multiplexers Mp. Figure 4 shows the circuit of the switch SQG(7, 4, 2) in this format. The (σ – 1)-channel fault tolerance is understood as the presence of σ different paths through different switches DSp between the input demultiplexers Dp and the output multiplexers Mp.

In the general case of a more complex network based on the switch SQG(N, p, σ), its (σ – 1)-channel fault tolerance is interpreted as the presence of σ different paths through different segments of the network between the input demultiplexers Dp and the output multiplexers Mp.

Unfortunately, quasi-complete graphs do not exist for all values of the parameters p and σ; for each pair

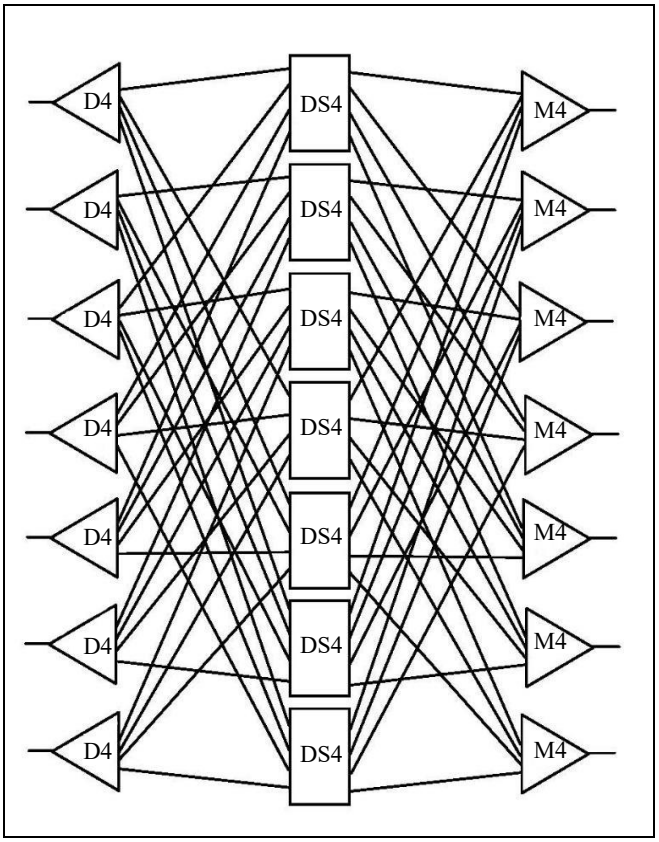


Fig. 4. Non-blocking self-routing switch SQG(7, 4, 2) with two different paths between any input demultiplexers D4 and output multiplexers M4.

of p and σ, they have to be constructed by enumeration. Table 3 presents the values of N quasi-complete graphs for small values of these parameters. Empty cells indicate graphs not existing by definition. The dashes in the cells correspond to the graphs that cannot exist according to the theory. Finally, the crossed-out values denote the graphs that have not yet been constructed.

Table 3

Parameters of switch SQG(N, p, σ)

	p									
σ	3	4	5	6	7	8	9	10	11	12
1	7	11	21	31	–	57	73	91	~~111~~	133
2	4	7	11	16	–	–	37	–	~~56~~	–
3	3	5		11	15		25	31	~~45~~	

The need to construct fault-tolerant networks requires some effective filling of empty cells in Table 3. For this purpose, one-extended switches SQG($N^*, p, \sigma|\sigma + 1$) were constructed in [25], in which a small part of the users are connected through (σ + 1) differ-

ent paths, and the rest through exactly σ different paths. The numbers N and N^* of nodes in the above-mentioned block designs are given in Table 4. (The latter numbers are underlined.)

The switches SQG(N_1, p, σ) have one layer of output multiplexers Mp, $V_1 = (p + 1)N_1$ in total.

Table 4

Switch SQG(N, p, σ) and one-extended block designs of switches SQG (N^*, p, σ|σ + 1)

	p									
σ	3	4	5	6	7	8	9	10	11	12
1	7	11	21	31	<u>39</u>	57	73	91	<u>95</u>	133
2	4	7	11	<u>15</u>	<u>21</u>	<u>27</u>	37	<u>42</u>	<u>51</u>	<u>63</u>
3	3	5	<u>7</u>	11	15	<u>19</u>	<u>23</u>	<u>29</u>	<u>36</u>	<u>43</u>

4. A NON-BLOCKING SELF-ROUTING TWO-STAGE FAULT-TOLERANT SWITCH

We introduce the notion of p-partitions of packets transmitted through some cross-section of a network under a permutation. All packages are divided into groups of variable composition, each containing at most p packages. For a common permutation of packets, a 1-partition occurs at the input and output of the network. A transmission in which a 1-partition occurs at the network input and a p-partition on some cross-section will be called a direct p-permutation. A transmission in which a p-partition occurs at the network input and a 1-partition on a given cross-section will be called an inverse p-permutation.

For the dual switch SQG(N_1, p, σ), this cross-section is through the inputs of the output multiplexers and called the output cross-section. According to the property of the dual switch DSp, a p-partition occurs on the output cross-section of the dual switch SQG(N_1, p, σ) for any traffic. In particular, a direct p-permutation occurs under any permutation in the dual switch SQG(N_1, p, σ).

Lemma 1. The dual switch SQG(N_1, p, σ) is non-blocking under any inverse p-permutation and preserves (σ – 1)-channel fault-tolerance.

P r o o f. The second assertion is based on the non-blocking property of the switch SQG(N_1, p, σ) on an arbitrary 1-permutation and the fact that an inverse p-permutation consists of sparse 1-permutations separated by different cycles. The number of different paths between any source and sink is preserved in each such permutation. ♦

Multichannel fault-tolerant networks will be further constructed on an example of the switches DS2 and SQG(2, 2, 2) (the minimal switch with single-channel fault-tolerance, see Fig. 5). First, a blocking two-stage four-channel $N_2$4 network with exchange links is constructed, as shown in Fig. 6. It turns out to be blocking due to possible conflicts in the multiplexers M2 of the first stage, shaded in the figure. One-channel fault tolerance is also violated on them. With two copies of the second stage and internal parallelization, the network $N_2$4 is transformed into a non-blocking self-routing switch $S_2$4 (Fig. 7).

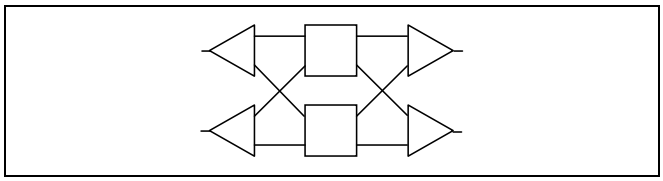


Fig. 5. Two-channel non-blocking switch SQG(2, 2, 2) with $T_2 = 2$ and one-channel fault tolerance.

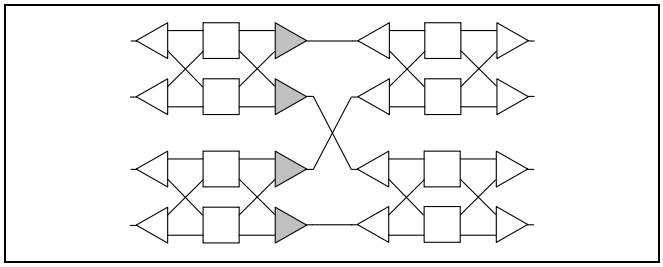


Fig. 6. Two-stage network $N_2$4 with exchange links.

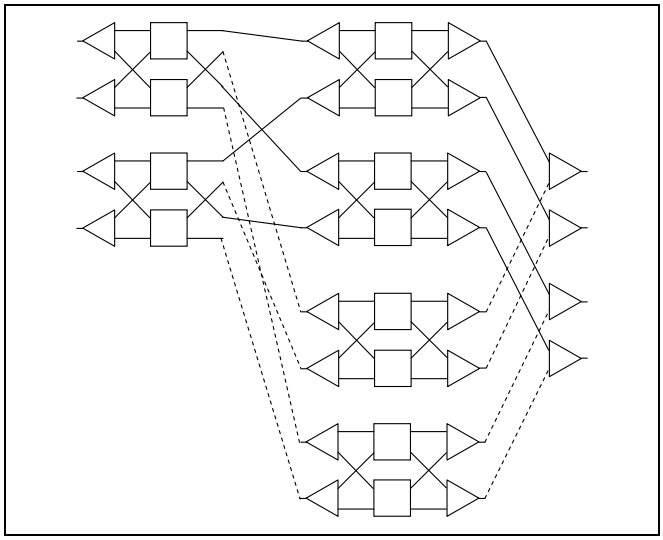


Fig. 7. Four-channel non-blocking self-routing switch $S_2$4 with one-channel fault tolerance.

This transformation is carried out by removing all multiplexers M2 from the first stage and dividing their input channels between two copies of the second stage. The cut multiplexers M2 combine the same-named output channels of the copies of the second stage, forming the first dimension circuit. Sparse direct p-permutations with disjoint inputs-outputs occur at the inputs of its switches SQG(2, 2, 2). By Lemma 1, they are implemented without conflict. In addition, any two paths on the first stage are divided between the two copies of the second stage, which preserves one-channel fault tolerance.

As a result, the switch $S_2 4$ is non-blocking with static self-routing. It has one-channel fault tolerance: all paths between the input demultiplexers D2 and the output multiplexers M2 pass through different copies of the second stage of the network $N_2 4$ in the first dimension circuit since $p = \sigma = 2$.

In the general case ($p > 2$ and $\sigma = 2$), the switches DSp and SQG(N_1, p, 2) are used. The minimal non-trivial dual graph of the switch SQG(N_1, p, 2) is the dual switch SQG(4, 3, 2); see Fig. 8.

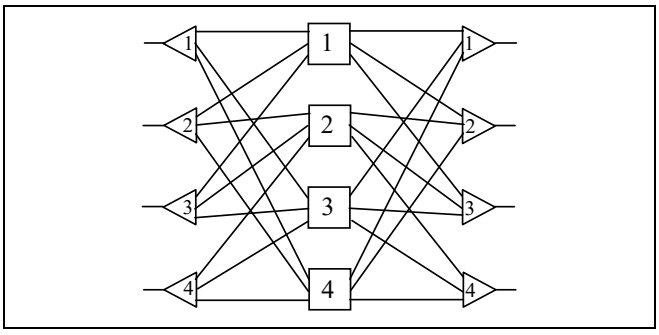


Fig. 8. Non-blocking self-routing dual switch SQG(4, 3, 2) with one-channel fault tolerance.

The two-stage network $N_2 N_2$ with exchange links and $N_2 = N_1^2$ is constructed by analogy. Then it is transformed into a non-blocking self-routing switch $S_2 N_2$ using internal parallelization over p copies of the second stage, which form the first dimension circuit. The switch $S_2 N_2$ is one-channel fault-tolerant: it has p paths in the first dimension circuit, and $p \geq 2$. For $p = 3$, the switch $S_2 16$ is shown in Fig. 9.

The switching complexity of a dual $p \times p$ switch has the formula $S_0 = 2p^2$. Then the switching complexity of the switch SQG(N_1, p, σ) is given by $S_1 = N_1 S_0 + 2pN_1$; its channel complexity, by $L_1 = 2pN_1$.

The switch $S_2 N_2$ uses all the multiplexers of the switches SQG(N_1, p, σ) without adding new ones, and their input and output channels are redirected only.

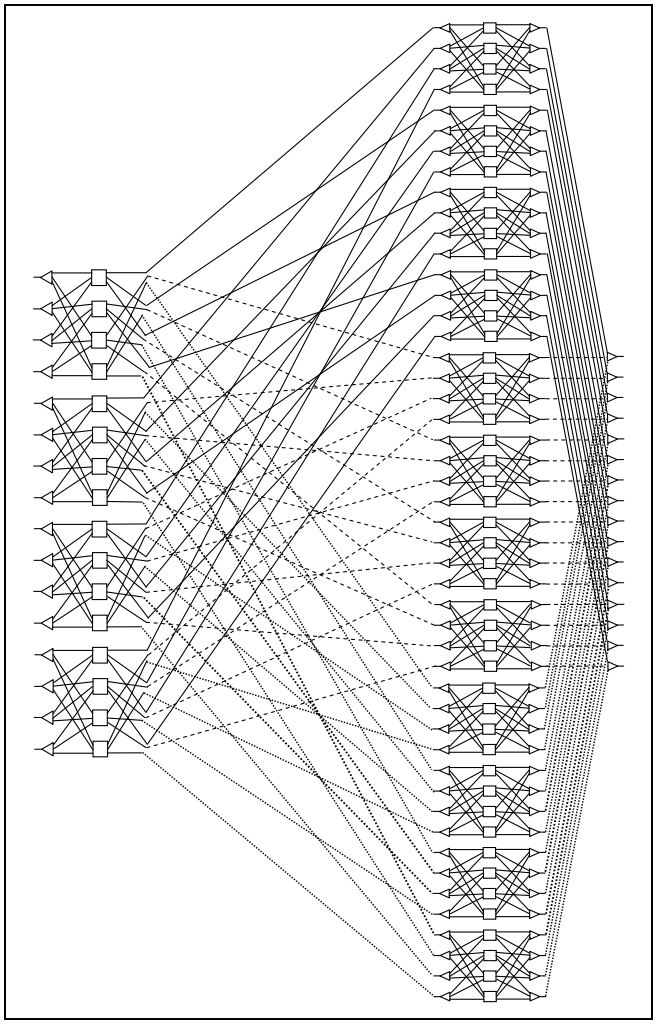


Fig. 9. Non-blocking self-routing switch $S_2 16$ with one-channel fault tolerance.

Hence, we conclude that by construction, the switch $S_2 N_2$ has the switching complexity $S_2 = N_1 S_1 + pN_1 S_1$ and the channel complexity $L_2 = N_1 L_1 + pN_1 L_1$. The performance characteristics of the switch $S_2 N_2$ for different σ are presented in Tables 5 and 6.

We compare the complexities of the switch $S_2 N_2$ with one-channel fault tolerance and a switch with the complete graph topology and duplicated channels, e.g., for $p = 4$ ($N_2 = 49$). The latter switch has the switching complexity $S_2 = N_2^{2.35}$ and the channel complexity $L_2 = N_2^{2.17}$. Obviously, their switching complexities are commensurable, but the channel complexity of the switch $S_2 N_2$ is significantly smaller. In addition, the digit period of the switch with the complete graph topology is four times smaller. However, it seems too early to finalize the conclusions.

Generally, the first dimension circuit is defined as follows: p copies of the second stage of the network $N_2 N_2$, with the inputs of the cut multiplexers Mp connected to them and the outputs combined by these multiplexers Mp, form the first dimension circuit.

Table 5

Performance characteristics of switches S_2N_2 with one-channel fault tolerance

p	N_1	$N_2 = N_1^2$	$T_2 = p$	S_2	L_2
2	2	4	2	$144 = N_2^{3.58}$	$56 = N_2^{2.9}$
3	4	16	3	$1536 = N_2^{2.65}$	$432 = N_2^{2.19}$
4	7	49	4	$9800 = N_2^{2.37}$	$2156 = N_2^{1.97}$
5	11	121	5	$43\,560 = N_2^{2.23}$	$7865 = N_2^{1.87}$
6	15	225	6	$132\,300 = N_2^{2.18}$	$20\,250 = N_2^{1.84}$
7	21	441	7	$395\,136 = N_2^{2.12}$	$52\,479 = N_2^{1.79}$
8	27	729	8	$944\,784 = N_2^{2.09}$	$110\,808 = N_2^{1.77}$

Table 6

Performance characteristics of switches S_2N_2 with two-channel fault tolerance

p	N_1	$N_2 = N_1^2$	$T_2 = p$	S_2	L_2
3	3	9	3	$864 = N_2^{3.08}$	$243 = N_2^{2.50}$
4	5	25	4	$5000 = N_2^{2.65}$	$1100 = N_2^{2.18}$
5	7	49	5	$17\,640 = N_2^{2.52}$	$3185 = N_2^{2.07}$
6	11	121	6	$71\,148 = N_2^{2.31}$	$10\,890 = N_2^{1.94}$
7	15	225	7	$201\,600 = N_2^{2.26}$	$26\,775 = N_2^{1.88}$
8	19	361	8	$467\,856 = N_2^{2.22}$	$54\,872 = N_2^{1.85}$

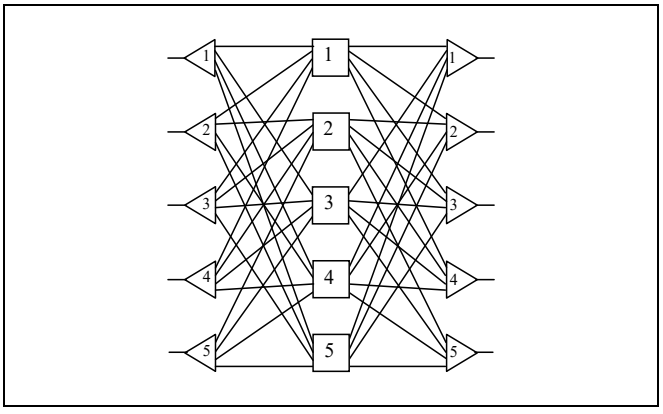


Fig. 10. Non-blocking self-routing dual switch SQG(5, 4, 3) with two-channel fault tolerance.

The switch S_2N_2 contains only one circuit of the first dimension, whose outputs are the outputs of the switch itself.

To provide two-channel fault tolerance, we use graphs of SQG(N_1, p, 3). The minimal graph of SQG(N_1, p, 3) is SQG(3, 3, 3), and the minimal nontrivial graph is SQG(5, 4, 3); see Fig. 10.

In the general case ($p \geq 3$ and $\sigma = 3$, see Table 4), the network N_2N_2 with $N_2 = N_1^2$ is constructed by analogy. Then, as described above, it is transformed into a non-blocking self-routing switch S_2N_2 using internal parallelization with p copies of the second stage. The switch S_2N_2 has two-channel fault tolerance.

We compare the complexities of the switch S_2N_2 with two-channel fault tolerance and a switch with the complete graph topology and tripled channels, e.g., for $p = 4$ ($N_2 = 25$). The latter switch has the switching complexity $S_2 = N_2^{2.55}$ and the channel complexity $L_2 = N_2^{2.34}$. Obviously, their switching complexities are commensurable, but the channel complexity of the switch S_2N_2 is significantly smaller. In addition, the digit period of the switch with the complete graph topology is four times smaller.

We draw a given cross-section in the switch S_2N_2 after the dual switches SQG(N_1, p, σ), i.e., at the input of the output multiplexers layers. Then the following lemma holds for any values p and σ.

Lemma 2. The dual switch S_2N_2 has a direct p-permutation on the indicated cross-section. It is a non-blocking switch with static routing on any reverse p-permutation for any p and has (σ – 1)-channel fault tolerance.

P r o o f. The first assertion is based on using the dual switch SQG(N_1, p, σ) and Lemma 1 for it. The second assertion is based on the non-blocking property of the switch SQG(N_1, p, σ) and the fact that an inverse p-permutation consists of sparse 1-permutations separated by different cycles. The union of the sparse alternative permutations through the cut multiplexers Mp generates no conflicts.

The property of (σ – 1)-channel fault tolerance follows from the fact that all paths between the input demultiplexers

Dp and the output multiplexers Mp pass through p copies of the second stage of the network N_2N_2 in the first dimension and $p \geq \sigma$. On the other hand, only σ different paths come from any source in the original dual switch SQG(N_1, p, σ), and their failure will break the network connectivity. ♦

CONCLUSIONS

This paper has proposed a method for constructing a new type of non-blocking self-routing photon networks in the form of a two-stage switch with $(\sigma - 1)$-channel fault tolerance. The new switch differs from those proposed in the authors' previous works by a new property called channel fault tolerance. This property has been achieved by replacing the basic dual switch with the quasi-complete digraph topology from [5] with a dual switch with the quasi-complete graph topology. Due to such a replacement, two or three communication channels are physically implemented between each "input–output" pair, in contrast to one channel in the original photon switches [1–4]. Each additional channel can be used in the event of a failure of the main channel. With the increase in the fault tolerance of photon switches, their channel and switching complexities have grown expectedly. This paper has presented expressions for calculating both performance characteristics depending on the number of channels.

In addition, the proposed designs of fault-tolerant two-stage switches are poorly scaled by external invariant parallelization with additional multiplexers and demultiplexers [5], which was used in [4] to scale a two-stage switch without channel fault tolerance based on the quasi-complete digraph topology. For example, the switch $S_2$16 with one-channel fault tolerance (Fig. 12) can be extended using multiplexers M3 and demultiplexers D3 only up to the switch $S_2$20 and, using the repeated extension, to the switch $S_2$25.

Therefore, the following problem arises to improve photon networks of large dimensions: construct a non-blocking self-routing fault-tolerant multistage switch of high scalability. This problem can be solved by extending such a switch into four-stage and eight-stage non-blocking switches based on the development and application of a generalized internal parallelization method.

Note that the exponential switching complexity can be significantly reduced by increasing the number of network stages and using a generalized internal parallelization method for each stage. This approach to scaling fault-tolerant photon switches is a continuation of this study and will be considered separately.

REFERENCES

1. Barabanova, E.A., Vytovtov, K.A., and Podlazov, V.S., Multistage Switches for Optical and Electronic Supercomputer Systems, *Proceedings of the 8th National Supercomputer Forum* (*NSCF–2019*), *Pereslavl-Zalessky*, 2019. URL: http://2019.nscf.ru/TesisAll/02_Apparatura/037_BarabanovaEA.pdf. (In Russian.)
2. Barabanova, E.A., Vytovtov, K.A., Vishnevsky, V.M., and Podlazov, V.S., The New Principle for the Construction of Optical Information Processing Devices for Information-Measuring Systems, *Sensors and Systems*, 2019, no. 9, pp. 3–9. (In Russian.)
3. Barabanova, E., Vytovtov, K., Podlazov, V., and Vishnevskiy, V. Model of Optical Non-blocking Information Processing System for Next-generation Telecommunication Networks, *Proceedings of the 22nd International Conference on Distributed Computer and Communication Networks: Control, Computation, Communications* (*DCCN–2019*), Moscow, 2019. Communications in Computer and Information Science, vol. 1141, Cham: Springer, pp. 188–198. DOI: 10.1007/978-3-030-36625-4_16.
4. Barabanova, E.A., Vytovtov, K.A., and Podlazov, V.S., Two-Stage Dual Photon Switches in an Extended Scheme Basis, *Control Sciences*, 2021, no. 1, pp. 69—81.
5. Karavai, M.F. and Podlazov, V.S., An Invariant Extension Method for System Area Networks of Multicore Computational Systems. An Ideal System Network, *Automation Remote Control*, 2010, vol. 71, no. 12, pp. 2644–2654.
6. Pipenger, N., On Rearrangeable and Non-Blocking Switching Networks, *J. Comput. Syst. Sci.*, 1978, vol. 17, pp. 307–311.
7. Scott, S., Abts, D., Kim, J. and Dally, W., The Black Widow *High-radix Clos Network, Proc. 33rd Intern. Symp. Comp. Arch.* (*ISCA'2006*), Boston, 2006. https://www.researchgate.net/publication/4244660_The_Black Widow_High-Radix_Clos_Network.
8. Gu, Q.P., and Tamaki, H., Routing a Permutation in Hypercube by Two Sets of Edge-Disjoint Paths, *J. of Parallel and Distributed Comput.*, 1997, vol. 44, no. 2, pp. 147–152.
9. Lubiw, A. Counterexample to a Conjecture of Szymanski on Hypercube Routing, *Inform. Proc. Let*., 1990, vol. 35(2), pp. 57–61.
10. Alverson, R., Roweth, D., and Kaplan, L., The Gemini System Interconnect, *Proc. 18th IEEE Symposium on High Performance Interconnects*, 2009, pp. 3–87.
11. Ajima,Y., Inoue, T., Hiramoto, S., and Shimiz, T., Tofu: Interconnect for the K Computer. URL: https://www.researchgate.net/publication/265227674_Tofu_Interconnect_for_the_K_computer.
12. Arimili, B., Arimili, R., Chung, V., et al., The PERCS High-Performance Interconnect, *Proc. 18th IEEE Symposium on High Performance Interconnects*, 2009, pp. 75–82.
13. De Sensi, D., Di Girolamo, S., McMahon, K.H., Roweth, D., and Hoefler, T., An In-Depth Analysis of the Slingshot Interconnect, *arXiv: 2008.08886v1*, August 20, 2020. URL: https://www.researchgate.net/publication/343786515_An_In-Depth_Analysis_of_the_Slingshot_Interconnect.
14. Stepanenko, S., Structure and Implementation Principles of a Photonic Computer, *EPJ Web of Conferences 224*, vol. 04002, 2019. DOI: https://doi.org/10.1051/epjconf/201922404002

15.Babaee, A., Momeni, A., Abdolali, A., and Fleury, R., Parallel Optical Computing Based on MIMO Metasurface Processors with Asymmetric Optical Response, *Phys. Rev. Applied*, 2021, vol. 15, article no. 044015.

16.Cheng, Q., Glick, M., and Bergman, K., Optical Interconnection Networks for High Performance Systems, in *Optical Fiber Telecommunications*, vol. VII, 2019, pp. 785–825.

17.Maniotis, P., Dupuis, N., Kuchta, D.M., Taubenblatt, M.A., and Lee, B.G., Intra-Node High-Performance Computing Network Architecture with Nanosecond-Scale Photonic Switches, *Journal of Optical Communications and Networking*, 2020, vol. 12(12), pp. 367–377.

18.Duro, J., Petit, S., Góme, M.E., and Sahuquillo, J., Segment Switching: A New Switching Strategy for Optical HPC Networks, *IEEE Access*, 2021, vol. 9, pp. 43095–43106.

19.Lee, B.G., Dupuis, N., Pepeljugoski, P., et al., Silicon Photonic Switch Fabrics in Computer Communications Systems, *Journal of Lightwave Technology*, 2015, vol. 33(4), pp. 768–777.

20.Krishnamoorthy, A.V., Ho, R., Zheng, X., et al., Computer Systems Based on Silicon Photonic Interconnects, *Proceedings of the IEEE*, 2009, vol. 97(7), pp. 1337–1361.

21.Cheng, Q., Rumley, S., Bahadori, M., and Bergman, K., Photonic Switching in High Performance Datacenters, *Optics Express*, 2018, vol. 26(12), article no. 16022.

22. Meyer, H., Sancho, J.C., Mrdakovic, M., et al., Optical Packet Switching in HPC. An Analysis of Applications Performance, *Future Generation Computer Systems*, 2018, vol. 82, pp. 606–616.

23.Seok, T.J., Kwon, K., Henriksson, J., et al., Wafer-Scale Silicon Photonic Switches beyond Die Size Limit, *Optica*, 2019, vol. 6(4), pp. 490–494.

24.Hall, M., *Combinatorial Theory*, Waltham: Blaisdell Publishing Company, 1967.

25.Karavai, M.F. and Podlazov, V.S., Expanded Block-Diagrams for Ideal System Area Networks, *Control Sciences*, 2012, no. 4, pp. 45–51. (In Russian.)

This paper was recommended for publication by V.G. Lebedev, a member of the Editorial Board.

Received March 11, 2021, and revised May 13, 2021.
Accepted May 13, 2021.

Author information

Barabanova, Elizaveta Aleksandrovna. Dr. Sci. (Eng.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, ✉ elizavetaalexb@yandex.ru

Vytovtov, Konstantin Anatol'evich. Dr. Sci. (Eng.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, ✉ vytovtov_konstan@mail.ru

Podlazov, Viktor Sergeevich. Dr. Sci. (Eng.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, ✉ podlazov@ipu.ru

Cite this article

Barabanova, E.A., Vytovtov, K.A., Podlazov, V.S. Non-blocking Fault-Tolerant Two-Stage Dual Photon Switches. *Control Sciences* **4,** 67–76 (2021). http://doi.org/10.25728/cs.2021.4.7

Original Russian Text © Barabanova, E.A., Vytovtov, K.A., Podlazov, V.S., 2021, published in *Problemy Upravleniya*, 2021, no. 4, pp. 82–92.