ISSN 2782-2427

CONTROL SCIENCES

4/2024

[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, RAS Academician

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: October 7, 2024

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

4.2024

CONTENTS

DOI: http://doi.org/10.25728/cs.2024.4.1

ROBUST CONTROLLER DESIGN ENSURING THE DESIRED APERIODIC STABILITY DEGREE OF A CONTROL SYSTEM WITH AFFINE UNCERTAINTY

S. A. Gayvoronskiy*, I. V. Khozhaev**, and A. V. Sobol***

*,**,***National Research Tomsk Polytechnic University, Tomsk, Russia

*✉ saga@tpu.ru, **✉ ivh1@tpu.ru, ***✉ avs127@tpu.ru

Abstract. This paper considers a system whose characteristic polynomial coefficients are linear combinations of the interval parameters of a plant forming a parametric polytope. A linear robust controller is parametrically designed to place a dominant pole of the system within the desired interval of the negative real semi-axis and ensure an aperiodic transient in the system. The parametric design procedure involves a low-order controller with dependent and free parameters: the former serve to place the dominant pole within the desired interval on the complex plane whereas the latter to shift the other poles to some localization regions beyond a given bound (to the left of the dominant pole to satisfy the pole dominance principle). To evaluate the dependent parameters of the controller, the originals of the interval bounds of the dominant pole are determined for the plant's parametric polytope based on a corresponding theorem (see below). The free parameters of the controller are chosen using the robust vertex or edge *D*-partition method, depending on the boundary edge branches of the localization regions of the free poles. A numerical example of the parametric design procedure is provided: a PID controller is built to ensure an acceptable aperiodic transient time in a load-lifting mechanism with interval values of cable length and load weight.

Keywords: robust control, affine uncertainty, modal control, aperiodic transient.

INTRODUCTION

As is known, ensuring a given performance of an automatic control system (ACS) is possible based on the desired placement of its poles, implemented by the modal controller design using the characteristic polynomial of the system. In such cases, the most frequently solved problem is to ensure an aperiodic transient of a given duration in the system. With aperiodic transients in the system, one decreases the amount of energy to bring the plant to the desired state as well as reduces the wear of the actuator. Let us consider known methods for solving this problem for systems with deterministic parametric uncertainty [1–29]. It is logical to classify the design approaches described therein by the type of controller and the order of the plant. In particular, linear controllers with constant parameters [1–10], adaptive controllers [12–14], controllers based on fuzzy logic [15–17], and neural network controllers [18–20] were applied. In turn, linear plants of given orders or nonlinear plants with the linear part of a given order were considered in [4–10]; no constraints on the order of the plant or its linear part were imposed in the other works. A criterion for the aperiodicity of systems with interval parameters was given in [11].

According to the analysis of the publications cited above, an aperiodic transient is most often ensured using linear controllers of different structures whose order coincides with that of the plant or its linear part: common controllers in the classical or modified form and polynomial controllers. In such cases, it is possible to derive symbolic expressions for calculating the controller parameters [2, 4, 5, 8, 10]. The criterion presented in [11] verifies whether all poles of a system with interval parameters lie on the real axis. The desired placement of all poles requires controllers whose order depends on the number of system poles. The implementation of full-order controllers is often com-

plicated due to the impossibility of directly measuring the derivatives of the output, which are necessary to form the control signal.

In addition, there are known methods for designing linear low-order controllers, which have no constraints on the order of the plant and are based on the pole dominance principle [21–23]. The disadvantage of these methods is the increased conservatism of the designed system due to the interval uncertainty of the characteristic polynomial coefficients.

Controllers based on fuzzy logic and neural network controllers were also designed for plants of arbitrary order [15–20]. However, such controllers are more difficult to implement than common linear controllers of both low and full order.

Thus, it is topical to design common linear controllers of low order that ensure an aperiodic transient of a given duration in systems with deterministic parametric uncertainty without restricting the order of the plant. To reduce the conservatism of the designed system, it is topical to consider the methods of placing the poles of systems with affine uncertainty of the coefficients of the interval characteristic polynomial.

If aperiodic transients of a given duration are required in an ACS, the pole dominance principle should be applied in the controller design: the real pole corresponding to the transient time is selected as the dominant one, and the other (free) poles are shifted on the left of it beyond some bound.

If the plant of the ACS has uncertain parameters whose values change during the system operation within specified intervals according to a priori unknown laws, there arises the problem of preserving aperiodic transients for any possible values of the interval parameters. When solving this problem, one should keep in mind that the system poles migrate inside their localization regions. Therefore, with the designed controller, the real dominant pole should be localized within some interval on the real negative semi-axis. At the same time, the localization regions of the free poles should be removed to a sufficient distance from this interval. Such localization of the poles ensures the completion of the ACS transients in an admissible time for any values of the interval parameters of the plant.

1. PROBLEM STATEMENT

For a time-invariant ACS, the pole placement principle (see above) can be implemented by a low-order controller based on the methodology proposed in [21]. According to it, the controller parameters are divided into dependent and free: the former set the dominant poles, whereas the latter shift all other poles to a certain region of the complex plane by the *D*-partition method.

Based on this approach, a methodology for designing a controller that ensures an aperiodic transient in a time-varying system with an interval characteristic polynomial was developed in [22, 23]. The coefficients of this polynomial are defined by limits found from the known intervals of the plant parameters and interval arithmetic rules. The coefficients form a parametric polytope, and the robust vertex *D*-partition method is applied at its vertices to select the free parameter of the controller.

However, the approach proposed in [21–23] allows an independent variation of the polynomial coefficients inside their polytope, making the resulting robust controller conservative. To reduce the conservatism of this approach, it is desirable to pass from the interval uncertainty of the characteristic polynomial to the affine one. Such a possibility exists if the polynomial coefficients are a linear combination of the interval parameters. In this case, the polytope of the interval parameters of the plant is considered instead of the polytope of the interval coefficients when designing the controller. It is mapped into the interval of the real dominant pole of the ACS and the localization regions of its free poles.

This paper aims to design a robust controller ensuring the aperiodic stability degree of an ACS with affine uncertainty based on determining, for the polytope of the plant parameters, the originals of the interval bounds of the system's dominant pole and the interval bounds of its free poles for their placement by the robust *D*-partition method [24].

2. MAPPING THE EDGES OF THE POLYTOPE OF THE INTERVAL PLANT PARAMETERS INTO THE ROOT PLANE

We write the characteristic equation of a system with interval parameters of the plant in the form

$$D(s) = \sum_{i=1}^{m} [T_i] A_i(s) + B(s) = 0, \qquad (1)$$

where $[T_i]$ are the interval parameters, $\underline{T_i} = \min(T_i)$, and $\overline{T_i} = \max(T_i)$. Let $A_i(s)$ be polynomials of s, which corresponds to the affine uncertainty of the polynomial (1). We denote by $B(s)$ the sum of the characteristic polynomial terms not containing the interval parameters.

Since the m interval parameters T_i are given by bounds, they will vary arbitrarily within a parametric polytope representing the rectangular hyperparallelepiped $P_T = \left\{T_i \mid \underline{T_i} \le T_i \le \overline{T_i},\ i = \overline{1,m}\right\}$ with 2^m verti-

ces. The coordinates of any point P_T relative to a vertex V_q, $q=\overline{1,2^m}$, are given by

$$T_i = T_i^q + \Delta T_i,\ i=\overline{1,m}, \qquad (2)$$

where ΔT_i is the increment of the i th parameter; T_i^q is the value of the i th parameter at the vertex V_q. Substituting the sum (2) into the expression (1) yields the equation

$$D^q(s) + \Delta T_1 A_1(s) + \Delta T_2 A_2(s) + ... + \Delta T_m A_m(s) = 0, \qquad (3)$$

where $D^q(s) = \sum_{i=1}^{m} T_i^q A_i(s) + B(s)$ is the polynomial at the vertex V^q. Based on formula (3), we write the mapping equation of an edge P_T outgoing from the vertex V^q when changing the value of the parameter T_i:

$$D^q(s) + \Delta T_i A_i(s) = 0. \qquad (4)$$

Using equation (4) and the root locus theory [4], we form the transfer function to construct the edge branch by the parameter T_i:

$$W_i^q(s) = \frac{\Delta T_i A_i(s)}{D^q(s)}. \qquad (5)$$

Due to the expression (5), the roots of the equation $\Delta T_i A_i(s) = 0$ are the zeros of the edge transfer function $W_i^q(s)$, and the roots of the vertex polynomial $D^q(s)$ are its poles.

3. THE INTERVALS OF REAL POLES OF THE SYSTEM WITH AFFINE UNCERTAINTY: SOME PROPERTIES

According to the root locus theory [25], the branches of a root locus are in definite parts of the real axis depending on the number of real zeros and poles of the system. For the interval extension of this property, it is necessary to determine, for the polytope of the interval parameters of the system, the vertices mapped into the interval bounds of the real poles. For this purpose, we establish the following result.

Proposition. *The right bound of the interval* $[s_j^L, s_j^R]$ *of the real pole* s_j *of the system with the interval parameters* T_i *is the image of the vertex* V_q *with the coordinates* $T_i^q = \underline{T_i}$ *provided that the total number of the intervals of the other system poles located to the right of* s_j^R *is even and the zeros of the edge transfer function* (5) *are constant. If the total number of the right intervals and zeros is odd, then* $T_i^q = \overline{T_i}$. *In this case, the coordinates of the vertex original of the left bound* s_j^L *have the opposite limits of the interval parameters.*

The proof of this proposition is provided in the Appendix.

Corollary 1. *The real pole* s_1 *determining the robust aperiodic stability degree of the system with the interval parameters* T_i *has the vertex original* V_q *with the coordinates* $T_i^q = \underline{T_i}$ *provided that the number of the real zeros of the edge transfer function* (5) *located to the right of* s_1 *is even. If the number of the real zeros is odd, then* $T_i^q = \overline{T_i}$.

Corollary 2. *If there are no real zeros of the edge transfer function* (5) *between the two intervals of the real poles of the system, then the right and left bounds of these intervals are associated with the vertices with the opposite limits of the interval parameters.*

A numerical example below illustrates the application of the proposition and its corollaries.

Example. Consider the characteristic polynomial of an ACS with affine uncertainty in which the polynomial coefficients are a linear combination of three interval parameters:

$$\begin{aligned}&[T_1]s^4 + (6[T_1]+[T_2])s^3 + (11[T_1]+5[T_2])s^2 \\ &+(6[T_1]+6[T_2]+[T_3])s+3[T_3]+1=0,\end{aligned} \qquad (6)$$

where $[T_1]=[5, 10]$, $[T_2]=[30, 70]$, and $[T_3]=[10, 20]$. We transform the polynomial (6) to (1):

$$[T_1]A_1(s)+[T_2]A_2(s)+[T_3]A_3(s)+1=0, \qquad (7)$$

where $A_1(s)=s^4+6s^3+11s^2+6s$, $A_2(s)=s^3+5s^2+6s$, and $A_3(s)=s+3$.

For this fourth-degree polynomial, the multiparametric interval root locus is represented by four intervals on the negative real semi-axis: $s_1=[-0.38, -0.07]$, $s_2=[-1.92, -1.44]$, $s_3=[-3.03, -3.01]$, and $s_4=[-15, -4.1]$. These intervals are shown in Fig. 1 (for clarity, without precise scale), together with the real roots of the polynomials $A_i(s)$ (7), namely:

– the roots of $A_1(s)$: $s_1=0$, $s_2=-1$, $s_3=-2$, and $s_4=-3$;

– the roots of $A_2(s)$: $s_1=0$, $s_2=-2$, and $s_3=-3$;

– the root of $A_3(s)$: $s_1=-3$.

Based on the mutual arrangement of the intervals and these roots, by the proposition, we obtain the coordinates of the vertices mapped into the bounds of the root intervals. Obviously, these vertices coincide with those determined when constructing the intervals of the real roots of the interval characteristic polynomial (Fig. 1).

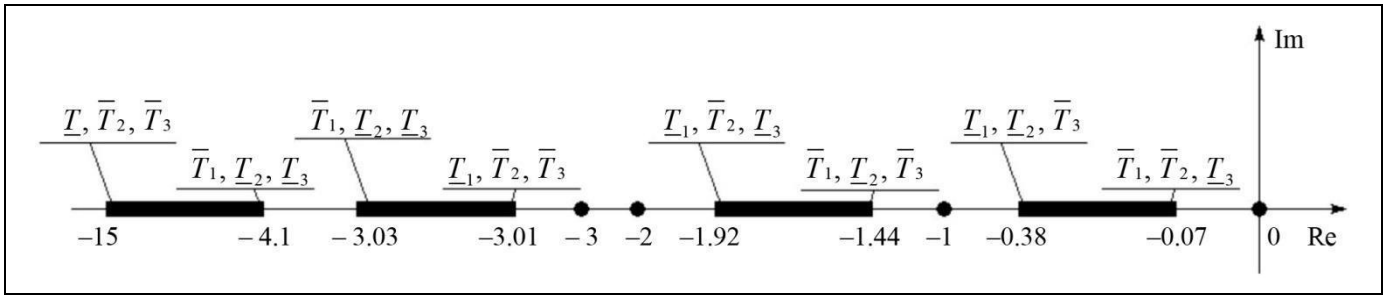

Fig. 1. Interval bounds for the real roots of the characteristic polynomial and the coordinates of their vertex originals.

4. FREE POLES LOCALIZATION BY THE ROBUST *D*-PARTITION METHOD

When changing one of the plant's interval parameters ΔT_i in the corresponding interval, the characteristic polynomial roots form a one-parameter interval root locus according to the root locus theory. The branches of this locus are called edge branches (RS_i^q), and their beginnings and ends are called root nodes (U_q). In addition, the edges and vertices of the parametric polytope are mapped into the branches and root nodes of the root locus, respectively: $\varphi(R_i^q) = RS_i^q$ and $\varphi(V_q) = U_q$.

By the edge theorem [26, 27], the localization regions of the roots of a characteristic polynomial with the affine interval uncertainty of its coefficients are bounded by the edge branches RS_i^q, i.e., the images of definite edges of the plant's parametric polytope. Following [23], we consider an edge branch RS_i^q formed by the motion of a complex conjugate root. If one of its ends is nearest to the imaginary axis, then the branch belongs to the first type; if one of its inner roots is nearest to the imaginary axis, and the original of this root is a priori unknown, then the branch under consideration belongs to the second type. To determine the type of an edge branch, we should check the following condition [28]: if the polynomial $A_i(s)$ under the interval uncertain parameter T_i is a polynomial of degree one, or of only an even degree, or of only an odd degree (of the variable s), or the product of such polynomials, then the edge branch RS_i^q belongs to the first type.

In the problem under study, the type of the edge branches bounding the localization regions of free poles is important for the robust *D*-partition [24] by the free parameter of the controller with a selected boundary of these regions. For instance, if all branches belong to the first type, then it suffices to perform *D*-partition at all vertices of the plant's parametric polytope and choose the value of the free parameter from the intersection of the regions obtained for each vertex.

If the above condition fails for some branch, then this branch belongs to the second type. In this case, it is necessary to apply *D*-partition by two parameters: the free parameter of the controller and the parameter of the corresponding branch of the polytope edge. (The coefficients of the characteristic polynomial must be a linear combination of these parameters.) By drawing the known bounds of the interval parameter in the parametric region, we obtain the admissible region of the controller's free parameter ensuring the desired placement of the free poles. After obtaining such regions for each branch of the second type, it is necessary to find their intersection and choose the value of the free parameter from it.

5. A NUMERICAL EXAMPLE: PID CONTROLLER DESIGN FOR A CABLE TENSION STABILIZATION SYSTEM OF A LOAD-LIFTING MECHANISM

Using the proposition above, we design a robust controller placing the dominant pole of a system within the desired interval and the other poles in a given region.

For this purpose, we consider the automatic cable tension stabilization system of a load-lifting mechanism [29]; see the block diagram in Fig. 2.

The notations are as follows: ΔF_{in} and ΔF_{out} are the increments of the cable tension force at the system input and output, respectively; DCM is a direct current motor; PA is a power amplifier.

The characteristic polynomial of the system has the form

$$a_5 s^5 + a_4 s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0 = 0,$$

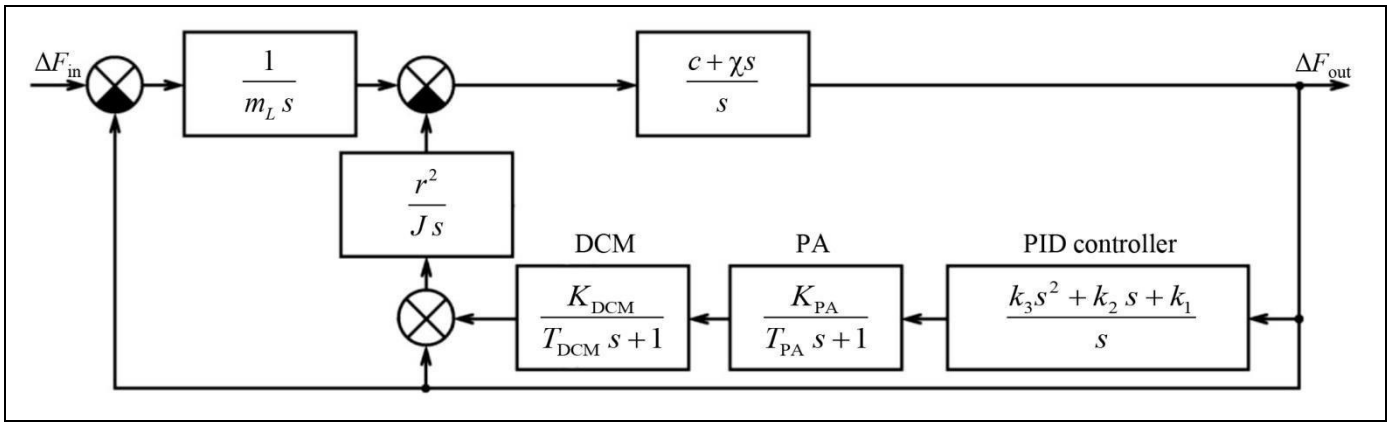

Fig. 2. The block diagram of the cable tension stabilization system of the load-lifting mechanism.

where $a_5 = J\,T_{DCM}\,T_{PA}\,l\,m_L$, $a_4 = T_{DCM}\,T_{PA}\,\chi\,m_L\,r^2 + J\,T_{DCM}\,T_{PA}\,\chi + JT_{DCM}\,lm_L + JT_{PA}\,l\,m_L$, $a_3 = Jlm_L + JT_{DCM}\chi + JT_{PA}\chi + JT_{DCM}T_{PA}c + T_{DCM}\chi\,m_L r^2 + T_{PA}\,\chi\,m_L r^2 + T_{DCM}T_{PA}c\,m_L r^2 + k_3 K_{DCM}K_{PA}\chi\,m_L r^2$, $a_2 = J\chi + \chi\,m_L\,r^2 + J\,T_{DCM}\,c + J\,T_{PA}\,c + T_{PA}\,c\,m_L\,r^2 + T_{DCM}\,c\,m_L\,r^2 + k_3 K_{DCM}K_{PA}c\,m_L r^2 + k_2\,K_{DCM}\,K_{PA}\,\chi\,m_L\,r^2$, and $a_1 = Jc + c\,m_L\,r^2 + k_1 K_{DCM}K_{PA}\,\chi\,m_L r^2 + k_2\,K_{DCM}\,K_{PA}\,c\,m_L\,r^2$, $a_0 = k_1\,K_{DCM}\,K_{PA}\,c\,m_L\,r^2$; m_L is the load mass, in kg; l is the cable length, in m; J is the moment of inertia of the electric drive, in $\mathrm{kg\cdot m^2}$; χ is the specific damping coefficient of the cable, in $\mathrm{N\cdot s}$; c is the specific stiffness of the cable, in N; r is the radius of the drive pulley of the electric drive, in m; K_{DCM} is the gain of the DC motor, in $\mathrm{rad\cdot s^{-1}\cdot V^{-1}}$; T_{DCM} is the time constant of the DC motor, in s; K_{PA} is the gain of the PA; finally, T_{PA} is the time constant of the PA, in s. The constant parameters of the plant have the following values: $J = 0.5\ \mathrm{kg\cdot m^2}$, $\chi = 10^4\ \mathrm{N\cdot s}$, $c = 2\cdot 10^4\ \mathrm{N}$, $r = 0.1\ \mathrm{m}$, $K_{DCM} = 5\ \mathrm{rad\cdot s^{-1}\cdot V^{-1}}$, $T_{DCM} = 0.01\ \mathrm{s}$, $K_{PA} = 10$, and $T_{PA} = 0.001\ \mathrm{s}$. The interval parameters of the plant are $m_L = [50, 500]\ \mathrm{kg}$ and $l = [50, 100]\ \mathrm{m}$. Thus, the parametric polytope of the system has four vertices.

The system uses a PID controller with the transfer function

$$W_{PID}(s) = \frac{k_3\,s^2 + k_2\,s + k_1}{s},$$

where k_1, k_2, and k_3 are the integral, proportional, and differential gains of the PID controller, respectively. We divide the controller parameters into dependent and free: the dependent parameters k_1 and k_3 determine the position of the dominant pole bounds whereas the free one k_2 the position of the other poles to the left of a given bound. Let it be required to localize the dominant pole within the interval $[-0.7, -0.5]$ of the real axis and to place the other poles to the left of the vertical line passing through the point $(-1, j0)$ parallel to the imaginary axis.

We substitute the constant parameters of the system into the characteristic polynomial, writing it as

$$l\,A_1(s) + \frac{1}{m_L}\,A_2(s) + A_3(s) = 0,$$

where $A_1(s) = s^3\,(5\cdot 10^{-10}\,s^2 + 5.5\cdot 10^{-7}\,s + 5\cdot 10^{-5})$, $A_2(s) = s(5\cdot 10^{-6}\,s^3 + 5.51\cdot 10^{-3}\,s^2 + 0.511s + 1)$, $A_3(s) = 10^{-7}\,s^4 + (0.5k_3 + 1.102\cdot 10^{-4})\,s^3 + (0.5k_2 + k_3 + 0.0102)\,s^2 + (0.5\,k_1 + k_2 + 0.02)\,s + k_1$.

Let us determine the coordinates of the vertex originals for the bounds of the real dominant pole of the system. The polynomial $A_1(s)$ has five roots: $s_1 = s_2 = s_3 = 0$, $s_4 = -100$, and $s_5 = -1000$. The polynomial $A_2(s)$ has four roots: $s_1 = 0$, $s_2 = -2$, $s_3 = -100$, and $s_4 = -1000$. Therefore, by the proposition above, the right bound $s^R = -0.5$ of the dominant pole is the projection of the parametric polytope vertex with the coordinates $(\bar{l};\ \underline{m_L})$; the left bound $s^L = -0.7$ of the dominant pole is the projection of the vertex $(\underline{l};\ \overline{m_L})$. Substituting the vertex coordinates and the interval bounds into the characteristic polynomial of the system yields two algebraic equations for the PID controller gains. By solving these equations, we obtain the following expressions for the dependent parameters of the controller:

$$\begin{cases} k_1(k_2) = 0.292\,k_2 + 0.025 \\ k_3(k_2) = 0.833\,k_2 - 0.017. \end{cases}$$

Note that $k_2 \geq 0.0206$ for a physically implementable controller.

Let us evaluate the free parameter k_2 of the controller using the *D*-partition method. Due to the form of the polynomials, the edge branches by the interval parameters of the system belong to the first type. Therefore, the ends of the edges are nearer to the imaginary axis than any of its interi-

or points. Consequently, it suffices to perform *D*-partition at the vertices of the parametric polytope to locate the free poles of the system. To obtain the equations of the *D*-partition curves, we substitute the values $k_1(k_2)$ and $k_3(k_2)$ as well as the bound $s = -1 + j\omega$ of the free poles into the characteristic polynomial of the system. The *D*-partition curves are shown in Fig. 3.

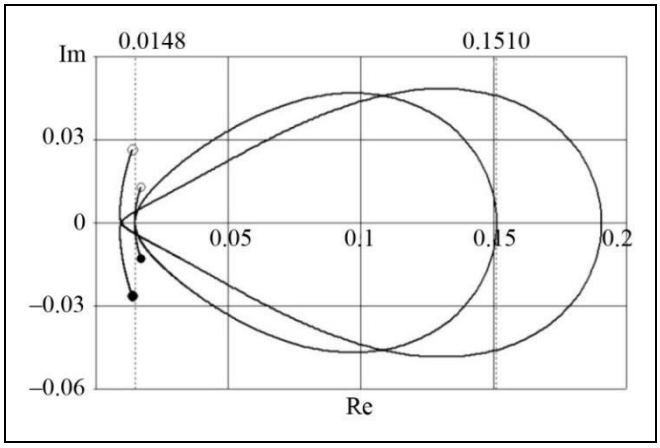

Fig. 3. The vertex *D*-partition curves by the free parameter of the controller.

Due to the *D*-partition, the desired pole placement is achieved for $k_2 \in [0.0148,\ 0.1510]$. Let us choose $k_2 = 0.1$. In this case, $k_1 = 0.054$ and $k_3 = 0.066$. The pole placement of the system with the designed PID controller is shown in Fig. 4.

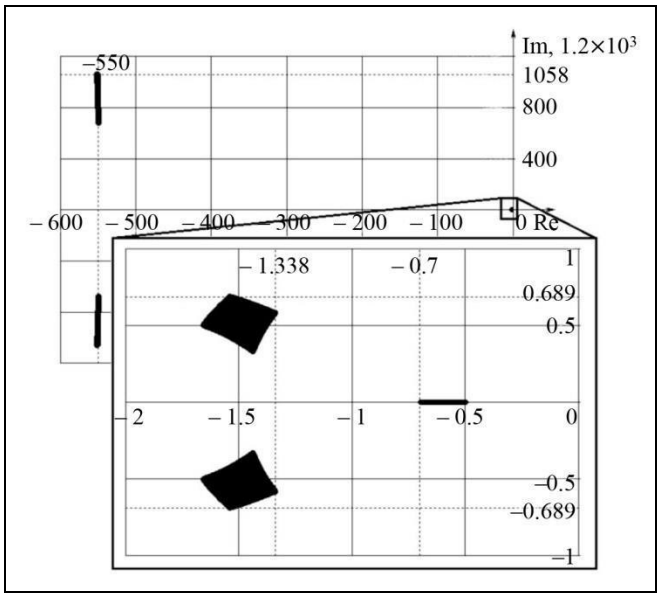

Fig. 4. The localization regions of the poles of the designed system.

With this placement, the design problem has been solved successfully: the real dominant pole is located within the desired interval, and the free poles are located to the left of the given bound. Next, Fig. 5 presents the family of transient characteristics of the designed system for different value combinations of the interval parameters.

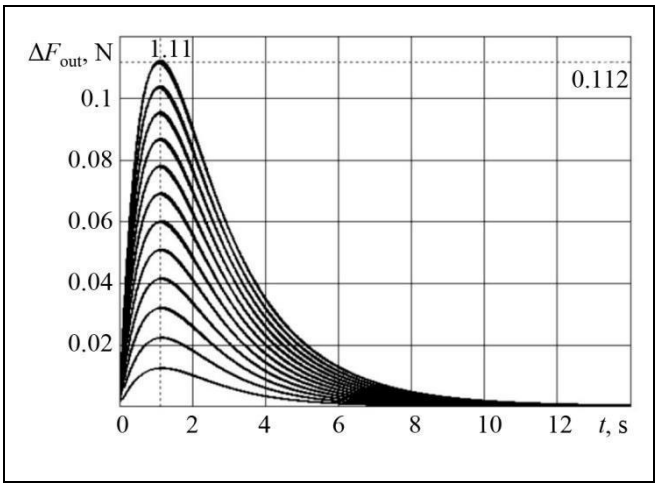

Fig. 5. The transient characteristics of the designed system.

According to Fig. 5, the designed controller ensures aperiodic transients in the system. The tension force occurring in the cable of length up to 100 m when suspending a load with mass up to 500 kg and weight up to 4900 N is compensated with a maximum deviation of 0.112 N or $2.2 \cdot 10^{-3}$%. Also, considering the transient characteristics in Fig. 5, the pole dominance principle holds despite a small distance between the free poles and the dominant one.

CONCLUSIONS

For technological reasons, the required aperiodic transient in an ACS often should be close to monotonic and contain, on the initial time interval, as little as possible oscillations from the free complex conjugate poles of the system. Therefore, to reduce the impact of the free poles on the transient, it is desirable to shift them from the interval of the real dominant pole to a sufficient distance defined by a given bound. If the shift cannot be performed using the *D*-partition method by one free parameter, then this method should be applied by two free parameters to expand the possibility of the desired localization of the free poles. For this purpose, if a typical PID controller with three parameters is used as a low-order controller without any constraints on the minimum transient time, one of the two dependent parameters of the controller should be made a free parameter. In this case, the remaining dependent parameter is used to ensure the given right bound of the dominant pole interval, which is enough to provide the maximum acceptable transient time in the ACS.

Acknowledgments. *This work was supported by the Russian Science Foundation, project no. 23-29-00737.*

APPENDIX

P r o o f the Proposition. The phase equation of the root locus has the form [25]

$$\sum_{i=1}^{m}\Theta_i^0-\sum_{i=1}^{n}\Theta_i=\pm(2n+1)\pi,$$

where Θ_i^0 is the angle of exit of the edge branch from the ith zero of the edge transfer function; Θ_i is the angle of exit of the edge branch from the ith pole of the edge transfer function; m is the number of zeros; n is the number of poles. Based on this equation, we write the equation of the angle Θ_1 of exit of the edge branch of the root locus from the right bound of the real pole s_j^R:

$$\Theta_1=r\pi-\sum_{i=1}^{m}\Theta_i^0+\sum_{i=2}^{n}\Theta_i,$$

where r is a parameter equal to 0 or 1 depending on the direction of motion along the edge branch of the root locus.

The angles in the equation depend on the location of the zeros and poles of the edge transfer function relative to the root locus point under study. Suppose that k of n poles and p of m zeros of the edge transfer function are located to the right of s_j^R. Let Θ_{Ri}^0 and Θ_{Ri} denote the angles corresponding to the right zeros and poles, and let Θ_{Li}^0 and Θ_{Li} denote the angles corresponding to the left zeros and poles. With these notations, the last equation takes the form

$$\Theta_1=r\pi-\left(\sum_{i=1}^{m-p}\Theta_{Li}^0+\sum_{i=1}^{p}\Theta_{Ri}^0\right)+\left(\sum_{i=1}^{n-k-1}\Theta_{Li}+\sum_{i=1}^{k}\Theta_{Ri}\right).$$

The zeros and poles of the edge transfer function can be either complex conjugate or real. Obviously, the angles corresponding to complex conjugate zeros or poles are equal in magnitude and opposite in sign. Consequently, when calculating the exit angle of an edge branch, they are mutually reduced and do not affect the final result. For the angles corresponding to the real zeros and poles, we have $\Theta_{Ri}^0=\Theta_{Ri}=\pi$ and $\Theta_{Li}^0=\Theta_{Li}=0$. Hence, the expression for Θ_1 can be written as

$$\Theta_1=r\pi-(m-p)\cdot 0-p\pi+(n-k-1)\cdot 0+k\pi=\pi(r+k-p).$$

The problem is to find the original V_q of the right bound of the real pole s_j^R. Consequently, it is necessary that $\Theta_{1i}=\pi$. To fulfill this condition, for even $(k-p)$ one should choose $r=1$ and, accordingly, $\underline{T_i}$ in the coordinates V_q. For odd $(k-p)$, one should choose $r=0$ and, accordingly, $\overline{T_i}$ in the coordinates V_q. Obviously, if the difference between the nonnegative integers k and p is even, then their sum is also even, and vice versa. Therefore, for the sake of convenience, when analyzing the placement of zeros and poles, we consider the sum of k and p, i.e., the total number of the zeros and poles of the edge transfer function located to the right of the point under study.

To find the original of the left bound s_j^L, it is necessary to set the exit angles of the edge branches as $\Theta_{1i}=0$. Hence, the values of the parameter r and the bounds of the interval parameters T_i in the coordinates V_q are set in opposite ways, and the originals of the right and left bounds of the real pole have opposite coordinates.

REFERENCES

1. Abbyasov, A.M. and Tararykin, S.V., Synthesis of Robust Control System for Transportation of Long-Length Material Based on Gramian Method, *Vestnik of Ivanovo State Power Engineering University*, 2023, no. 4, pp. 54–62. (In Russian.)
2. Kulikov, V.E., Inverse Problem of Modal Control Solution Using Reduced-Dimension Controller Synthesis for Aircraft Vertical Speed Hold, *Navigatsiya i Upravlenie Letatel'nymi Apparatami*, 2022, no. 38, pp. 39–59. (In Russian.)
3. Krasnoshchechenko, V.I., Synthesis Robust H_∞-Controller of the Low Order by Using of Linear Matrix Inequalities and Projective Lemmas, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2018, vol. 19, no. 4, pp. 219–231. (In Russian.)
4. Voronin, A.I. and Tyutikov, V.V., Procedure for the Synthesis of Controllers for Independent Formation of Static and Dynamic Parameters of Nonlinear Objects, *Izvestiya SFedU. Engineering Sciences*, 2015, no. 3 (164), pp. 154–164. (In Russian.)
5. Fokin, A.L., The Synthesis of Robust Control Systems of Technological Processes with Standard Rcontrollers, *Bulletin of St. Petersburg State Institute of Technology (Technical University)*, 2014, no. 27 (53), pp. 101–106. (In Russian.)
6. Frantsuzova, G.A. and Vostrikov, A.S., PID Controller Design for a Second-Order Nonlinear Plant, *Optoelectronics, Instrumentation, and Data Processing*, 2019, vol. 55, no. 4, pp. 364–370.
7. Frantsuzova, G.A., Robust Systems Synthesis with PI2D-controller for Nonlinear Objects with Variable Parameters, *Automatics & Software Enginery*, 2018, no. 2 (24), pp. 9–16. (In Russian.)
8. Tsavnin, A.V., Efimov, S.V., and Zamyatin, S.V., PID-Controller Tuning Approach Guaranteeing Non-overshooting Step Response, *Proceedings of TUSUR University*, 2019, vol. 22, no. 2, pp. 77–82. (In Russian.)
9. Rybin, I.A. and Rubanov, V.G., Synthesis of a Robust Controller for a Mobile Robot with Interval Parameters and Time Delay, *Proceedings of Irkutsk State Technical University*, 2017, vol. 21, no. 10 (129), pp. 40–52. (In Russian.)
10. Tatarinov, A.V. and Tsirlin, A.M., The Limiting Degree of Linear System Aperiodic Stability and the Choice of Industrial Controller Parameters, *Modeling and Analysis of Information Systems*, 2012, vol. 19, no. 2, pp. 87–96. (In Russian.)
11. Polyak, B.T. and Tsipkin, Ya.Z., Robust Aperiodicity, *Dokl. Math.*, 1994, vol. 39 (3), pp. 149–152.
12. Nikou, A., Verginis, C.K., and Heshmati-Alamdari, S., An Aperiodic Prescribed Performance Control Scheme for Uncertain Nonlinear Systems, *Proceedings of 2022 30th Mediterranean Conference on Control and Automation (MED)*, Athens, 2022, pp. 221–226.

13. Opeiko, O.F., Output Control with Adaptive-Proportional Differential Controller, *System Analysis and Applied Information Science*, 2016, no. 3, pp. 35–39. (In Russian.)
14. Zhou, X., Wang, Z., Shen, H., et al. Robust Adaptive Path-Tracking Control of Autonomous Ground Vehicles with Considerations of Steering System Backlash, *IEEE Transactions on Intelligent Vehicles*, 2022, vol. 7, no. 2, pp. 315–325.
15. Volosencu, C., Study of the Angular Positioning of a Rotating Object Based on Some Computational Intelligence Methods, *Mathematics*, 2022, vol. 10, no. 7, art. no. 1157.
16. Zhu, J. and Nguang, S.K., Fuzzy Model Predictive Control with Enhanced Robustness for Nonlinear System via a Discrete Disturbance Observer, *IEEE Access*, 2020, vol. 8, pp. 220631–220645.
17. Chen, M., Lamb, H.K., Xiao, B., and Xuan, C., Membership-Function-Dependent Control Design and Stability Analysis of Interval Type-2 Sampled-Data Fuzzy-Model-Based Control System, *IEEE Transactions on Fuzzy Systems*, 2021, vol. 30, no. 6, pp. 1614–1623.
18. Ghafoor, A. and Balakrishnan, S.N., Design and Analysis of Event-Triggered Neuro-Adaptive Controller (ETNAC) for Uncertain Systems, *Journal of the Franklin Institute*, 2020, vol. 357, no. 10, pp. 5902–5933.
19. Pajchrowski, T. and Zawirski, K., Application of Artificial Neural Network to Robust Speed Control of Servodrive, *IEEE Transactions on Industrial Electronics*, 2007, vol. 54, no. 1, pp. 200–207.
20. Li, J., Xiang, X., and Yang, S., Robust Adaptive Neural Network Control for Dynamic Positioning of Marine Vessels with Prescribed Performance under Model Uncertainties and Input Saturation, *Neurocomputing*, 2022, vol. 484, pp. 1–12.
21. Vadutov, O.S. and Gaivoronskii, S.A., Solving the Problem of Allocation of Poles of a System by the D-partition Method, *Journal of Computer and Systems Sciences International*, 2004, vol. 43, no. 5, pp. 681–685.
22. Khozhaev, I.V., Adaptive Robust Stabilization of an Aperiodic Transient Process Control Quality in Systems with Interval Parametric Uncertainty, *IFAC-PapersOnLine*, 2018, vol. 51, no. 32, pp. 826–831.
23. Khozhaev, I.V., Gayvoronskiy, S.A., and Ezangina, T.A., Adaptive-Robust Stabilization of Interval Control System Quality on a Base of Dominant Poles Method, *Control Sciences*, 2019, no. 6, pp. 22–31. (In Russian.)
24. Uderman, E.G., *Metod kornevogo godografa v teorii avtomaticheskikh sistem* (The Root Locus Method in Automatic Systems Theory), Moscow: Nauka, 1972. (In Russian.)
25. Bartlett, A.C., Hollot, C.V., and Lin, H., Root Locations of an Entire Polytope of Polynomials: It Suffices to Check the Edges, *Mathematics of Control, Signals and Systems*, 1988, vol. 1, no. 1, pp. 61–71.
26. Vicino, A., Robustness of Pole Location in Perturbed Systems, *Automatica*, 1989, vol. 25, no 1, pp. 109–113.
27. Zhabko, A.P. and Kharitonov, V.L., Necessary and Sufficient Conditions for the Stability of a Linear Family of Polynomials, *Automation and Remote Control*, 1994, vol. 50, no. 10, pp. 1496–1503.
28. Petrov, N.P. and Polyak, B.T., Robust D-partition, *Automation and Remote Control*, 1991, vol. 52, no. 11, pp. 1513–1523.
29. Gayvoronskiy, S.A. and Ezangina, T.A., The Synthesis of the Robust Stabilization System of Cable Tension for the Test Bench of Weightless Simulation, *Aerospace MAI Journal*, 2015, vol. 22, no. 1, pp. 67–74. (In Russian.)

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

Received April 1, 2024,
and revised August 13, 2024.
Accepted August 26, 2024.

Author information

Gayvoronskiy, Sergei Anatol'evich. Cand. Sci. (Eng.), National Research Tomsk Polytechnic University, Tomsk, Russia
✉ saga@tpu.ru
ORCID iD: https://orcid.org/0000-0002-7156-2807

Khozhaev, Ivan Valer'evich. Cand. Sci. (Eng.), National Research Tomsk Polytechnic University, Tomsk, Russia
✉ ivh1@tpu.ru
ORCID iD: https://orcid.org/0000-0002-8874-0200

Sobol, Aleksandr Vasil'evich. Postgraduate, National Research Tomsk Polytechnic University, Tomsk, Russia
✉ avs127@tpu.ru
ORCID iD: https://orcid.org/0009-0002-9327-4811

Cite this paper

Gayvoronskiy, S.A., Khozhaev, I.V., and Sobol, A.V., Robust Controller Design Ensuring the Desired Aperiodic Stability Degree of a Control System with Affine Uncertainty. *Control Sciences* **4**, 2–9 (2024). http://doi.org/10.25728/cs.2024.4.4

Original Russian Text © Gayvoronskiy, S.A., Khozhaev, I.V., Sobol, A.V., 2024, published in *Problemy Upravleniya*, 2024, no. 4, pp. 3–12.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences,
Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com

DOI: http://doi.org/10.25728/cs.2024.4.2

INVESTIGATION OF TANDEM QUEUING SYSTEMS USING MACHINE LEARNING METHODS

V. M. Vishnevsky*, A. A. Larionov**, A. A. Mukhtarov***, and A. M. Sokolov****

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

*✉ vishn@inbox.ru, **✉ larioandr@gmail.com,
✉ mukhtarov.amir.a@gmail.com, *✉ aleksandr.sokolov@phystech.edu

Abstract. This paper considers tandem queuing systems with limited buffer sizes in each phase. The system handles an incoming correlated MAP flow and the service time obeys a PH-distribution. Models of such systems and methods for their investigation are briefly reviewed from the historical perspective. According to the review, the problem statement presented below, the methods proposed for solving this problem, and the corresponding results are novel. An accurate algorithm for calculating the performance characteristics of low-dimensional tandem queuing systems is described, including an estimate of the algorithm's complexity. An approach using both machine learning and simulation modeling is suggested for the investigation of high-dimensional tandem queuing systems. Numerical analysis results are provided to show the effectiveness of machine learning methods for estimating the performance of tandem queuing systems.

Keywords: tandem queuing system, analytical model, simulation modeling, machine learning.

INTRODUCTION

Tandem queuing systems are commonly used to model and optimize the performance of various complex systems, such as technical, economic, industrial, transportation, medical, military, and others [1–4]. Of considerable interest are also the models of tandem queuing systems that adequately describe the operation of modern broadband wireless networks with linear topology [5, 6].

Since the late 1960s [7, 8], tandem queuing systems have been intensively studied up to the present time [9–11]. Initially, analytical results were obtained for two-phase tandem queuing systems. Many authors created methods for studying systems with blocking, incoming Poisson and correlated (Batch Markovian Arrival Process, BMAP) flows, and different distribution functions of the service time; for example, see [12–16]. More detailed descriptions of the works on two-phase systems can be found in the review [17] and the book [18].

However, tandem queuing systems of high dimensionality (with two or more phases) have the greatest practical importance. An analytical solution for these systems has been found only for specific networks that meet the conditions of the BCMP (Baskett–Chandy–Muntz–Palacios) theorem [19, 20] and have a multiplicative probability distribution. When the phases in a tandem network are M/M/1 queuing systems, one can easily calculate the system's steady-state performance characteristics, including the end-to-end delay as an important parameter. The end-to-end delay is the time of transmitting a packet from the first phase to the last.

For other types of high-dimensional tandem networks, finding analytical solutions is practically impossible. Therefore, approximate methods are widely used to study them. A common method for estimating performance characteristics involves dividing a network into separate parts and analyzing their interactions. This analysis can then be used to investigate the performance of the entire system (see [21–24]). In recent years, machine learning methods have been effectively used to study tandem queuing systems [9, 25–27]. In particular, the performance characteristics of tandem queuing systems with an incoming Poisson flow, the exponential distribution of the service time, and multilinear phases $(M/M/S/\infty)$ were analyzed in [9] using artificial neural networks.

This paper presents a new algorithm for accurately calculating the steady-state performance characteristics of tandem networks with an incoming correlated MAP-flow and the phase-type (PH)-distribution of the service time in the system phases. We estimate the complexity of this algorithm and describe its advantages and numerical analysis limitations concerning the number of system phases. A combination of machine learning and simulation modeling methods is used to study large tandem queuing systems. This approach enables rapid calculation of the characteristics of high-dimensional tandems, facilitating the design of complex practical systems [9, 28, 29].

1. PROBLEM STATEMENT

In this paper, we examine a tandem queuing system with $N \geq 2$ phases and a limited buffer size M_i $(i = 1,..., N)$ of each phase. Packets income the system in a MAP-flow governed by a control process $\nu_t (t \geq 0)$. This process is a Markov chain with a state space $\{0,1,..., W\}$ and matrices $\mathbf{D}_0$ and $\mathbf{D}_1$. The matrix $\mathbf{D}_1$ describes the changes in the controlling Markov chain when a packet is generated (observable transitions); the matrix $\mathbf{D}_0$, the changes without packet generation (unobservable transitions). The matrix $\mathbf{D} = \mathbf{D}_0 + \mathbf{D}_1$ is an infinitesimal generator of the Markov chain ν_t. The packet arrival rate, denoted by λ, is the product $\lambda = \bar{\boldsymbol{\pi}}\mathbf{D}_1\bar{\mathbf{1}}$, where $\bar{\boldsymbol{\pi}}$ is the steady-state distribution vector of the process ν_t, representing the unique solution of the system of algebraic equations $\bar{\boldsymbol{\pi}}\mathbf{D} = \bar{\mathbf{0}}$, $\bar{\boldsymbol{\pi}}\bar{\mathbf{1}} = 1$. Throughout this paper, $\bar{\mathbf{1}}$ and $\bar{\mathbf{0}}$ stands for a column vector composed of ones and a row vector composed of zeros, respectively (of appropriate dimensions).

The packet service time at the ith phase obeys a PH-distribution with an irreducible representation $(\mathbf{S}_i, \bar{\boldsymbol{\tau}}_i)$, $i = 1,..., N$. Here, $\mathbf{S}_i$ is a square matrix of order V_i (the number of process states) and $\bar{\boldsymbol{\tau}}_i$ is a vector defining the probabilities of the initial process state. Thus, the service process at the ith phase is controlled by a Markov chain with a state space $\{1,..., V_i, V_i + 1\}$, where $V_i + 1$ is an absorbing state. More detailed information about the PH-distribution and MAP-flow can be found, e.g., in [18]. If a packet arrives when a buffer is full, it will be immediately discarded and considered lost, without being serviced.

In what follows, we analyze two versions of tandem queuing systems: the system with cross-traffic (along with the outgoing flow Z_i, the ith phase receives an extra MAP-flow X_i) and the system without cross-traffic (external traffic arrives only at the first phase; see Fig. 1.

The problem is to estimate the steady-state performance characteristics of the described tandem queuing systems, including the end-to-end delay and the probability of packet loss.

2. A PRECISE ALGORITHM FOR CALCULATING THE PERFORMANCE CHARACTERISTICS OF THE TANDEM QUEUING SYSTEM WITH AN INCOMING MAP-FLOW, PH-DISTRIBUTION OF SERVICE TIME, AND LIMITED BUFFER SIZES

Let us examine some properties and characteristics of the MAP/PH/1/M system. They can be utilized to develop a precise algorithm for computing the tandem queuing system. The key property of the MAP/PH/1/M system is closedness on the set of MAP-flows according to the following theorems [18].

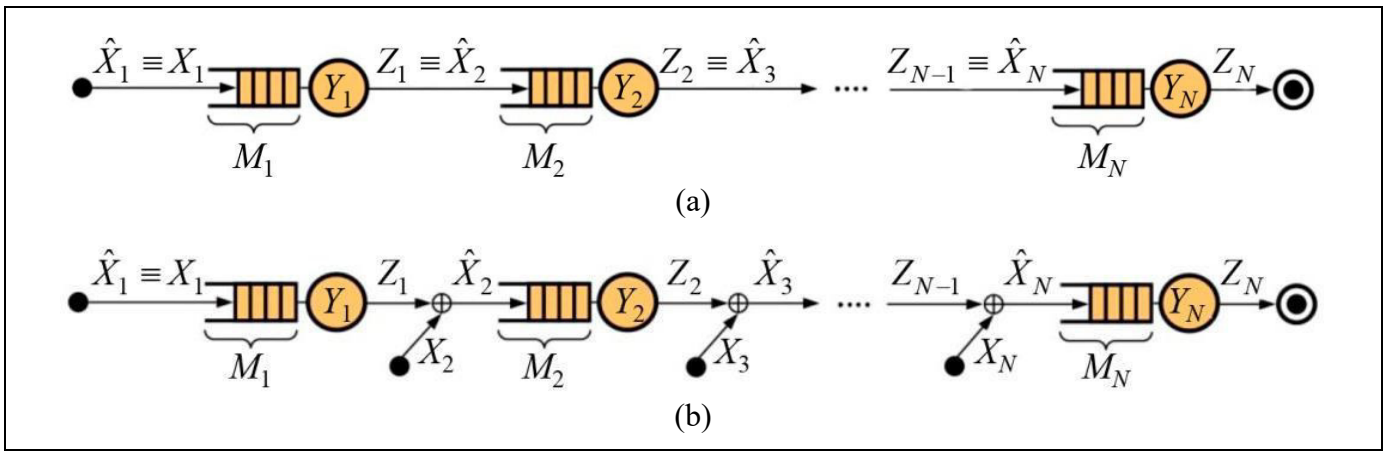

Fig. 1. Packet flows in the tandem queuing system: (a) with cross-traffic and (b) without cross-traffic.

Theorem 1. *The flow of serviced packets in the* MAP/PH/1/*M system, where the MAP-flow is given by X:* $MAP(\mathbf{D}_0, \mathbf{D}_1)$ *and the service time obeys the phase-type distribution Y:* $PH(\mathbf{S}, \bar{\boldsymbol{\tau}})$, *is an MAP-flow* $Z \sim MAP(\mathbf{D}'_0, \mathbf{D}'_1)$ *characterized by the matrices*

$$\mathbf{D}'_0 = \begin{bmatrix} \mathbf{D}_0 \otimes \mathbf{I}_V & \mathbf{D}_1 \otimes (\bar{\boldsymbol{\tau}} \otimes \bar{\mathbf{1}}_V) & 0 & \cdots & 0 & 0 \\ 0 & \mathbf{D}_0 \otimes \mathbf{S} & \mathbf{D}_1 \otimes \mathbf{I}_V & \cdots & 0 & 0 \\ 0 & 0 & \mathbf{D}_0 \otimes \mathbf{S} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \mathbf{D}_0 \otimes \mathbf{S} & \mathbf{D}_1 \otimes \mathbf{I}_V \\ 0 & 0 & 0 & \cdots & 0 & (\mathbf{D}_0 + \mathbf{D}_1) \otimes \mathbf{S} \end{bmatrix},$$

$$\mathbf{D}'_1 = \begin{bmatrix} 0 & \cdots & 0 & 0 & 0 \\ \mathbf{I}_W \otimes \mathbf{C}_t & \cdots & 0 & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & 0 \\ 0 & \cdots & 0 & \mathbf{I}_W \otimes \mathbf{C}_t & 0 \end{bmatrix},$$

where $\mathbf{C}_t = (-\mathbf{S}\bar{\mathbf{1}}_V) \otimes \bar{\boldsymbol{\tau}}$ *and* $\mathbf{I}_V$ *and* $\mathbf{I}_W$ *are identity matrices of order* V *and* W, *respectively.*

Theorem 2. *The superposition of MAP-flows* X_1 *and* X_2, $X_i \sim MAP(\mathbf{D}_0^{(i)}, \mathbf{D}_1^{(i)})$, $i = 1, 2$, *is a MAP-flow*

$$X = X_1 \oplus X_2 \sim MAP(\mathbf{D}_0^{(1)} \oplus \mathbf{D}_0^{(2)}, \mathbf{D}_1^{(1)} \oplus \mathbf{D}_1^{(2)}),$$

where $\oplus$ *denotes the Kronecker sum. If the flows* X_1 *and* X_2 *have orders* W_1 *and* W_2, *respectively, then the order of the total flow* X *is* $W = W_1 W_2$.

Let Z_i be the outgoing flow of the ith phase of the tandem queuing system and $\hat{X}_i$ be the total incoming flow of the ith phase (Fig. 1). According to Theorems 1 and 2, the flows $\hat{X}_i$ and Z_i are MAP-flows. Therefore, the ith phase can be described by a $\mathrm{MAP}_i / \mathrm{PH}_i / 1 / M_i$ queuing system with the packet arrival rate λ_i. For this system, the key performance characteristics are calculated by well-known formulas [18], including the average queue length m_i, the probability of packet loss $P_L^{(i)}$, the average end-to-end delay T_i of a packet at the ith phase, and others. By calculating these characteristics, we can determine the desired parameters for the probability of packet loss in the tandem queuing system,

$$P_L = 1 - \prod_{i=1}^{N} (1 - P_L^{(i)}),$$

and the end-to-end delay

$$T = \sum_{i=1}^{N} T_i = \sum_{i=1}^{N} \frac{m_1^{(i)}}{(1 - P_L^{(i)})\lambda_i}.$$

The next subsection presents a formal algorithm for calculating the steady-state performance characteristics of the tandem queuing system.

2.1. An Analytical Algorithm for Calculating the Steady-State Performance Characteristics of the Tandem Queuing System

Step 1. Set $i := 1$.

Step 2. If $i = 1$, set $\hat{X}_i = X_1$. In the case $i > 1$, calculate $\hat{X}_i$: $\hat{X}_i = Z_{i-1}$ if the system has no cross-traffic and $\hat{X}_i = Z_{i-1} \oplus X_i$ otherwise. Denote by $\hat{\mathbf{D}}_{i,0}$ and $\hat{\mathbf{D}}_{i,1}$ the matrices of the flow $\hat{X}_i$, i.e., $\hat{X}_i = \mathrm{MAP}(\hat{\mathbf{D}}_{i,0}, \hat{\mathbf{D}}_{i,1})$.

Step 3. Using Theorem 1, calculate the matrices $\mathbf{D}'_{i,0}, \mathbf{D}'_{i,1}$ of the MAP-flow $Z_i = \mathcal{D}(\hat{X}_i, Y_i, M_i)$.

Step 4. For the outgoing MAP-flow Z_i, determine the steady-state distribution $\bar{\boldsymbol{\theta}}^{(i)}$ using the system of linear algebraic equations

$$\begin{cases} \bar{\boldsymbol{\theta}}^{(i)} (\mathbf{D}'_{i,0} + \mathbf{D}'_{i,1}) & = 0 \\ \bar{\boldsymbol{\theta}}^{(i)} \bar{\mathbf{1}} & = 1. \end{cases}$$

Step 5. Calculate the average number of packets in the queue at the i th phase:

$$m_1^{(i)} = \sum_{k=0}^{M_i + 1} k \sum_{j=1}^{V_i \hat{W}_i} \theta_{kV_i \hat{W}_i + j}^{(i)},$$

where $V_i = |Y_i|$ is the order of the PH-distribution Y_i and $\hat{W}_i = |\hat{X}_i|$ is the order of the MAP-flow $\hat{X}_i$.

Step 6. Determine the steady-state probabilities $\bar{\boldsymbol{\pi}}^{(i)}$ of the incoming flow $\hat{X}_i$. If the system has no cross-traffic and $i > 1$, set $\bar{\boldsymbol{\pi}}^{(i)} \equiv \bar{\boldsymbol{\theta}}^{(i-1)}$. Otherwise,

find $\bar{\boldsymbol{\pi}}^{(i)}$ by solving the system of linear algebraic equations

$$\begin{cases} \bar{\boldsymbol{\pi}}^{(i)}(\hat{\mathbf{D}}_{i,0}+\hat{\mathbf{D}}_{i,1}) &= 0 \\ \bar{\boldsymbol{\pi}}^{(i)}\bar{\mathbf{1}} &= 1. \end{cases}$$

Step 7. Using the steady-state probabilities $\bar{\boldsymbol{\pi}}^{(i)}$ of the incoming MAP-flow $\hat{X}_i$ obtained at the previous step, calculate the arrival rate of packets at the *i*th phase:

$$\lambda_i = \bar{\boldsymbol{\pi}}^{(i)}\hat{\mathbf{D}}_{i,1}\bar{\mathbf{1}}.$$

Step 8. Determine the steady-state distribution of the incoming MAP-flow with M_i+1 packets in the system (i.e., when the system buffer is full):

$$\bar{\boldsymbol{\psi}}^{(i)} = \left(\sum_{j=1}^{V_i} \{\bar{\boldsymbol{\theta}}^{(i)}_{M_i+1}\}_j, \ldots, \sum_{j=1}^{V_i} \{\bar{\boldsymbol{\theta}}^{(i)}_{M_i+1}\}_{(\hat{W}_i-1)V_i+j} \right).$$

Here, the vector $\bar{\boldsymbol{\theta}}^{(i)}_{M_i+1}$ is the part of the vector $\bar{\boldsymbol{\theta}}^{(i)}$ relating to the system states when there are M_i+1 packets in the system.

Step 9. Calculate the probability of packet loss due to the buffer overflow of the *i*th phase:

$$P_L^{(i)} = \bar{\boldsymbol{\psi}}^{(i)}\frac{\hat{\mathbf{D}}_{i,0}}{\lambda_i}\bar{\mathbf{1}}.$$

Step 10. Calculate the average delay at the i th phase:

$$T_i = \frac{m_1^{(i)}}{(1-P_L^{(i)})\lambda_i}.$$

Step 11. If $i<N$, assign $i:=i+1$ and return to Step 2. Otherwise, proceed to Step 12.

Step 12. Calculate the probability of packet loss $P_L = 1-\prod_{i=1}^{N}(1-P_L^{(i)})$.

Step 13. Calculate the end-to-end delay (total delay) $T=\sum_{i=1}^{N}T_i$ of the tandem queuing system.

2.2. Estimating the Complexity of the Analytical Algorithm

The proposed scheme is computationally simple. At each step, block matrices are built for the outgoing MAP-flow using Kronecker product operations. In the case of cross-traffic in the system, matrices for the incoming MAP-flow are also constructed using the Kronecker sum. Next, one or two systems of linear algebraic equations are solved to determine the steady-state probabilities of the incoming and outgoing MAP-flow (the cases of cross-traffic and no cross-traffic, respectively). Finally, to calculate the probability of packet loss, the average system size, and the end-to-end delay, these steady-state distributions are multiplied by the flow matrices using several operations. The main disadvantage of this computation scheme is its extremely high computational complexity.

Proposition 1. *Assume that the incoming MAP-flows and the PH-distributions have orders W and V, respectively, the buffer size at each phase is M, and the system contains N phases. Then the complexity of the iterative scheme for calculating the performance characteristics of the tandem queuing system is estimated as follows:*

– $O((MVW)^{3N})$ *if the system has cross-traffic;*

– $O(W^3(MV)^{3N})$ *otherwise.*

P r o o f. Consider the i th iteration of the algorithm, where $i \leqslant N$, representing the computation of the performance characteristics of the i th phase of the system. Note that for $i>1$, the order of the outgoing MAP-flow from the preceding $(i-1)$th phase is $(M+2)V\hat{W}_i$, where $\hat{W}_i$ denotes the order of the incoming flow at the *i*th phase. If there is cross-traffic in the system, we have $U_i = ((M+2)VW)^i$; otherwise, $U_i = W((M+2)V)^i$.

The complexity of the iteration depends on Steps 4 and 6 (solving systems of linear algebraic equations). The system matrix at Step 4 (the outgoing flow generator) has a higher order than that at Step 6 (the incoming flow generator). Assuming that a Gaussian-like algorithm is used to solve the system, Step 4 requires $O(U_i^3)$ operations. The remaining steps have lower complexity: $O(1)$ for Steps 1, 10, and 11; $O(U_{i-1}^2W^2)$ for Step 2; $O(U_i^2)$ for Step 3; $O(VW+M)$ for Step 5; $O(U_i^2)$, for Steps 7 and 9; $O(VM)$. for Step 8. The complexity of Steps 12 and 13 is $O(N)$. ♦

Hence, if there is cross-traffic in the system, we obtain the algorithmic complexity

$$O((VWM)^3)+O((VWM)^6)+\ldots+O((VWM)^{3N}) +O(N) = O(VWM)^{3N};$$

in the case of no cross-traffic,

$$O(W^3(VM)^3)+O(W^3(VM)^6)+\ldots+O(W^3(VM)^{3N}) +O(N) = O(W^3(VM)^{3N}).$$

Thus, it is hard to find a solution using the algorithm even for relatively small *N*, *V*, and *W*. Table 1 provides the orders of the outgoing MAP-flows calculated under different system parameters. According to this table, a precise solution is feasible only for $N<5$

Table 1

Orders of the outgoing MAP-flows depending on the orders of the PH-distribution (*V*), MAP-flow of arrivals (*W*), and buffer size (*M*)

System parameters			Phase number				
W	*V*	*M*	1	2	3	4	5
Without cross-traffic							
1	1	1	3	9	27	81	243
1	1	3	5	25	125	625	3 125
2	2	2	16	128	1 024	8 192	65 536
3	1	3	15	75	375	1 875	9 375
1	3	3	15	225	3 375	50 625	759 375
3	3	3	45	675	10 125	151 875	2 278 125
With cross-traffic							
1	1	1	3	9	27	81	243
1	1	3	5	25	125	625	3 125
2	2	2	16	256	4 096	65 536	1 048 576
3	1	3	15	225	3 375	50 625	759 375
1	3	3	15	225	3 375	50 625	759 375
3	3	3	45	2 025	91 125	4 100 625	184 528 125

phases. More efficient computational schemes are required to apply high-dimensional tandem queuing systems with MAP/PH/1/*M* phases in practice.

3. ESTIMATING THE PERFORMANCE CHARACTERISTICS OF THE HIGH-DIMENSIONAL TANDEM QUEUING SYSTEM

As has been mentioned above, analytically calculating the performance of a tandem queuing system can be inefficient or impossible due to the high dimensionality of MAP-flows. The high computation time of the steady states and steady-state performance characteristics of the system has a considerable effect in iterative problems. For example, when designing a wireless communication network, it is important to select an optimal topology. This involves evaluating the characteristics and choosing the best option at each iteration step. To do it, we propose a new approach based on a combination of machine learning and simulation methods (Fig. 2). Within this approach, for different sets of input parameters, simulation modeling is used to generate a dataset with the calculated performance characteristics of the tandem queuing system. The generated dataset is then applied in a machine learning algorithm to obtain fast estimates of the performance characteristics. The approach was effectively used to solve problems of queuing theory [28–30]. This section describes the simulation model of the tandem queuing system as well as the procedures for validating this model using an analytical model and estimating the performance characteristics of the tandem queuing system using the combined method.

Calculating the steady-state characteristics of a tandem queuing system using the simulation method involves simulating the process of generating new packets and servicing them until a packet is lost or the packet service ends. Data on the generated and lost packets need to be stored. Average values for performance characteristics (phase delays and the probabilities of packet loss) should be calculated. The model is based on the discrete-event simulation principle: only emerging events (new packets generation and service completion) are processed. The time between successive events changes instantaneously: the model state remains invariable without event processing. During event processing, it is possible to calculate and assign the instants when new events will occur during model execution.

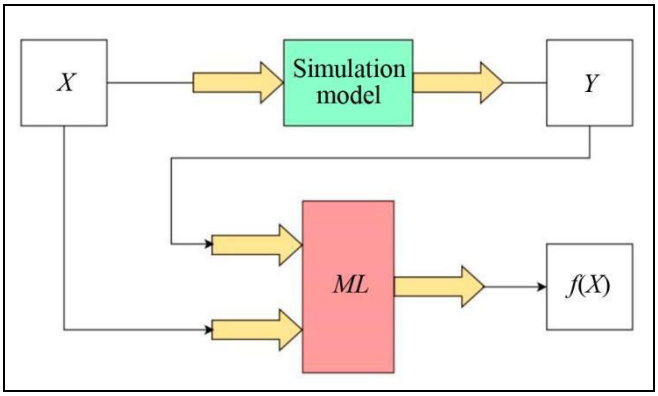

Fig. 2. The fast estimation procedure for the steady-state performance characteristics of the tandem queuing system.

3.1. Estimating the Performance Characteristics of the Tandem Queuing System by Simulation Modeling

Within this study, we developed a simulation model of the tandem queuing system in Python. Some modules of the simulation model were implemented in C++ to improve the performance and efficiency of calculations. Such modules were integrated into the model using Cython. The PyQumo library[1] was applied to implement and study stochastic models, in particular, multiphase queuing systems.

The simulation model has the following input parameters: the dimension N of the tandem queuing system (the number of phases), the buffer sizes $M_i, i = 1,...,N,$ of the phases; the incoming MAP-flow of packets at the first phase of the system, and the PH-

[1] URL: https://github.com/ipu69/pyqumo (Accessed September 16, 2024.)

distributions $PH_i, i = 1,...,N$, of the packet processing time at each phase. If certain phases are available, the incoming MAP cross-traffic flows are specified. If a buffer is empty, the packet goes directly to the server. Each phase receives packets from the previous phase, and the process continues until all packets are serviced at the last phase. If the packet is successfully processed at the last phase, it is considered to be successfully delivered; otherwise, the packet is considered to be lost.

The primary drawback of the simulation method is that the accuracy of the results strongly depends on the number of simulated events. For example, in a tandem queuing system with 10 phases, approximately 100 000 packets must be generated to achieve high accuracy with an error not exceeding 5%. Consequently, this method has limited potential for accelerating computation.

To validate the developed simulation model, we implemented the analytical algorithm for calculating the performance characteristics of the tandem queuing system (see Section 2) in Python. The simulation results were compared with those of the analytical calculations. Validation was performed on a set of 430 random networks, each consisting of 1 to 10 phases. The applicability of the precise algorithm is limited by the dimensionality of the flows of serviced packets (the input flows for the next phases), which grows for the nth phase as $(M+2)^N V^N W$. To ensure the correct operation of the model, the input datasets were generated so that the order of the outgoing flow at the last phase was below 8000.

For different performance characteristics of the tandem network, Fig. 3 shows the dependence of the relative error on the number of packets in the simulation model. The biggest error in the simulation model occurs when calculating the system size at the last phase. In this case, an error within 5% can be achieved by modeling only 25 000 packets; modeling 100 000 packets (see the dataset for training the regression model below) allows reducing the error to 1%.

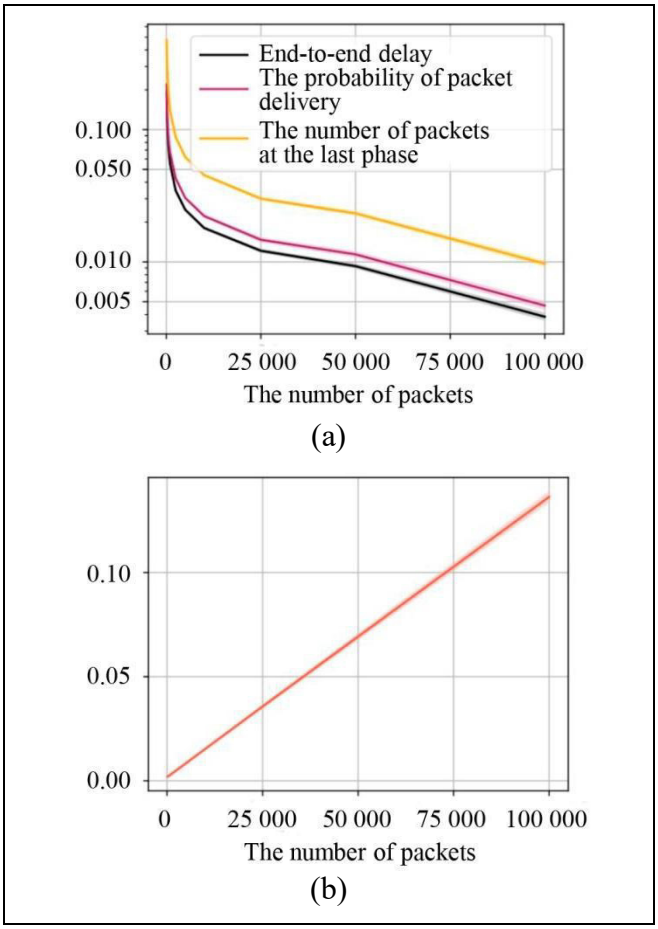

Fig. 3. The convergence of the simulation method and the speed of calculations: (a) relative error and (b) duration.

3.2. Estimating the Steady-State Performance Characteristics of the Tandem Queuing System by Machine Learning Methods

In this paper, we propose to use machine learning methods to accelerate the estimation of the performance characteristics of the tandem queuing system. The proposed methodology is effective in calculating the delay time between two phases and the probability of packet delivery in the tandem queuing system; see numerical examples in Section 4.

The following steady-state performance characteristics were calculated using the simulation model:

- the end-to-end delay in the tandem queuing system, Δt;
- the probability of packet delivery, $1 - P_L$.

Within this study, we built two models for predicting network performance characteristics. The first (regression) model estimates the average end-to-end delay, whereas the second model estimates the probability of successful packet delivery in the system. The models were built under the following constraints:

- the same buffer size for each phase;
- the same PH-distribution of the packet service time for each phase;
- an incoming MAP-flow with the zero autocorrelation coefficient.

Considering the restrictions, the following parameters were taken for the model: the first three moments of the MAP-flow, the first three moments of the PH-distribution, the tandem size (N), and the buffer size (M).

The first three moments were selected as features to characterize the incoming MAP-flow and the PH distributions of the service time. The original distributions can be reconstructed with these features. Let $m_a = \mathbb{E}X$ be the mean time between new packets arrivals and σ_a be its standard deviation. Similarly, let $m_s = \mathbb{E}Y$ be the average service time of a packet, and σ_s be its standard deviation. The service time is described by the mean m_s, the coefficient of variation $c_s = \sigma_s / m_s$, and the coefficient of skewness $\gamma_s = \mathbb{E}[(Y - m_s)^3] / \sigma_s^3$. By analogy, the incoming flow is described by m_a, c_a, and γ_a. As mentioned earlier, the autocorrelation coefficient for the incoming MAP-flow is supposed to be zero. In this case, the first three moments are enough to reconstruct the distributions of the time between packets arrivals and the service time.

In this paper, we employed two methods to reconstruct the distributions. First, we tried to build the second-order acyclic continuous PH-distribution in canonical form (ACPH(2); see Fig. 4a) using the method described in [31]. If the moments fall outside the existence region of ACPH(2), it is necessary to use the more universal method proposed in [32]: the PH-distribution is sought as a mixture of two Erlang distributions $\mathrm{ME}_n(2)$ (see Fig. 4b). In general, the PH-distribution can be found for any values $m > 0$, $c > 0$, and $\gamma > c - 1/c$ of a certain continuous positive distribution [32].

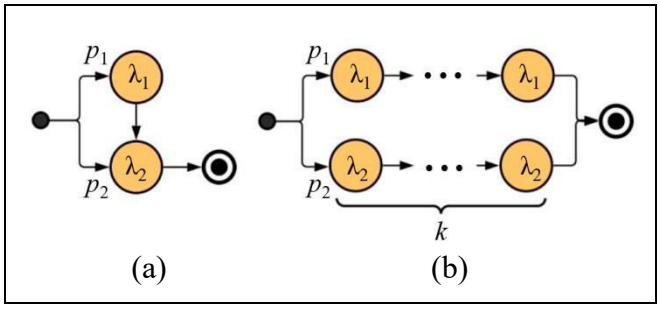

Fig. 4. (a) the acyclic PH-distribution with two ACPH(2) states and (b) the hyper-Erlang distribution with two Erlang distributions of the same order, used to approximate the flows by three moments.

Within this study, the following algorithms and methods were used to forecast the end-to-end delay: least squares method (LSM), tree-based algorithms (decision tree [33–36], gradient boosting [37]), and an artificial neural network with Adam's optimization algorithm [38]. Also, we applied classification models to estimate the probability of packet delivery, $1 - P_L$. Successful delivery was understood in the sense of exceeding a given value B from the interval $(0, 1)$. The algorithm determined the class, $[0, B)$ or $[B, 1]$, that the system belonged to. We used logistic regression, decision tree, gradient boosting, and artificial neural networks with Adam's optimization algorithm to solve the classification problem.

4. NUMERICAL RESULTS

Synthetic data generated by simulation modeling were used to estimate of the performance characteristics of the tandem queuing system. The simulation model received randomly generated input data in the ranges shown in Table 2. The outputs were the average delay time and the average probability of packet delivery, $\{\Delta t, 1 - P_L\}$. Simulation modeling yielded a huge dataset consisting of 101 424 rows of valuable information.

Table 2

Input parameters of the simulation model

Parameter	Range
Packet arrivals	
Mean, m_a	$\sim (0, 10)$
The coefficient of variation, c_a	$\sim (0.5,\ 3)$
The coefficient of skewness, γ_a	$\sim (c_a - \frac{1}{c_a}, 100)$
Service time	
Mean, m_s	$\sim (0{,}10)$
The coefficient of variation, c_s	$\sim (0.5, 3)$
The coefficient of skewness, γ_s	$\sim (c_s - \frac{1}{c_s}, 100)$
Buffer size, M	$\{6, 7, ..., 10\}$
The number of phases, N	$\{1, 2, ..., 20\}$

Since the data were generated randomly, the sample contains values of the loading coefficient $\rho = \frac{m_s}{m_a}$, strongly exceeding the range $(0, 1)$. It is unreasonable to use such data for further model training. Note that the system phases have a limited buffer size, so packets can be lost if a phase is overloaded. Thus, the load on the first phase may be $\rho >> 1$. Therefore, we restrict the analysis to the range $\rho \in (0, 10]$. After eliminating outliers, the sample size became 96 248 (rows).

4.1. Analysis of End-to-End Delays

We used the following metrics to assess the forecast values: the correlation coefficient (R), the

standard deviation (STD), and the coefficient of determination (R^2).

We analyzed the end-to-end delays estimated for different network dimensionalities, $N = 1, 5, 10, 20$. Figure 5 illustrates the architecture of an artificial neural network featuring a single hidden layer with 40 neurons. The sigmoidal activation function was selected, and training was conducted over 1000 epochs.

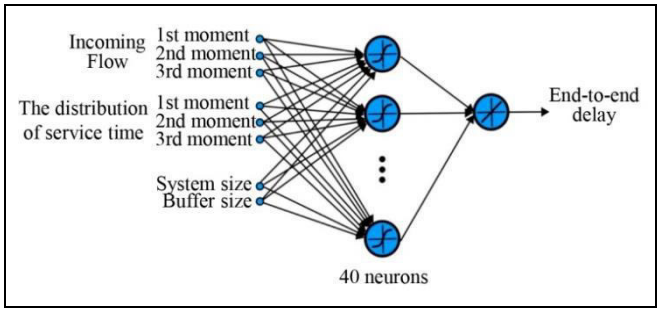

Fig. 5. The neural network architecture for forecasting the end-to-end delay.

The distribution densities of all end-to-end delay estimates in the test sample are presented in Fig. 6. Clearly, the least squares method produced the worst forecasting results; see the green curves in the graph. Due to the linear approximation, most of the estimates turned out to be negative. For the decision tree, the root-mean-square (RMS) error was taken as the partition criterion at each node of the tree. The best estimates were obtained for a tree depth of 36. For the gradient boosting method on the decision tree, a tree depth of 10 was chosen empirically. In the case under consideration, the learning rate was set equal to 0.1 and the number of trees equal to 100. Figure 7 shows the scatter diagrams of all trained models. According to Table 3, the neural network-based model demonstrated the best-quality forecasting.

4.2. Analysis of the Probability of Packet Delivery

Unlike the regression models for estimating the end-to-end delay, the delivery probability model does not need to forecast particular values. It is much more important to predict whether the delivery of packets will be successful or not. Let B be a given threshold for the successful delivery condition. We classified the probabilities of packet delivery into two groups: $1 - P_L \in [B, 1]$ (successful delivery) and $P_L \in [0, B)$ (packet loss). In the numerical experiment, we took the threshold $B = 0.9$ for all models. The models were assessed in terms of the following metrics: Precision, Recall, and F_1.

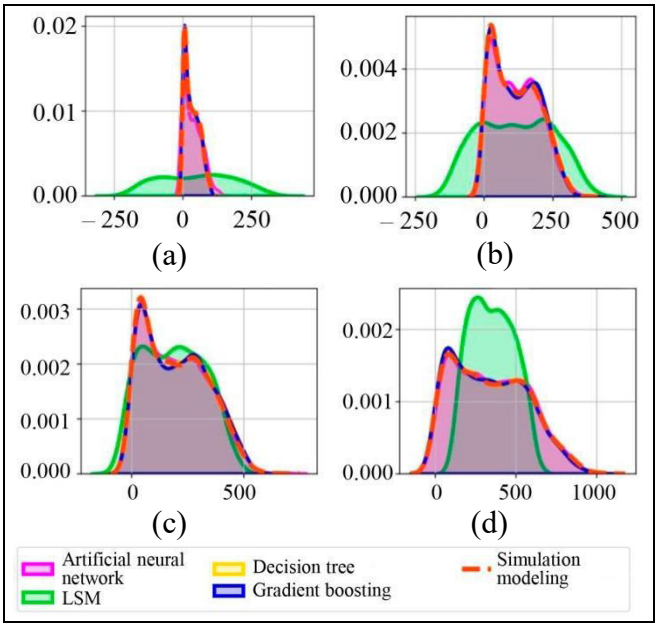

Fig. 6. The distribution density of the end-to-end delay: (a) network size 1, (b) network size 5, (c) network size 10, and (d) network size 20.

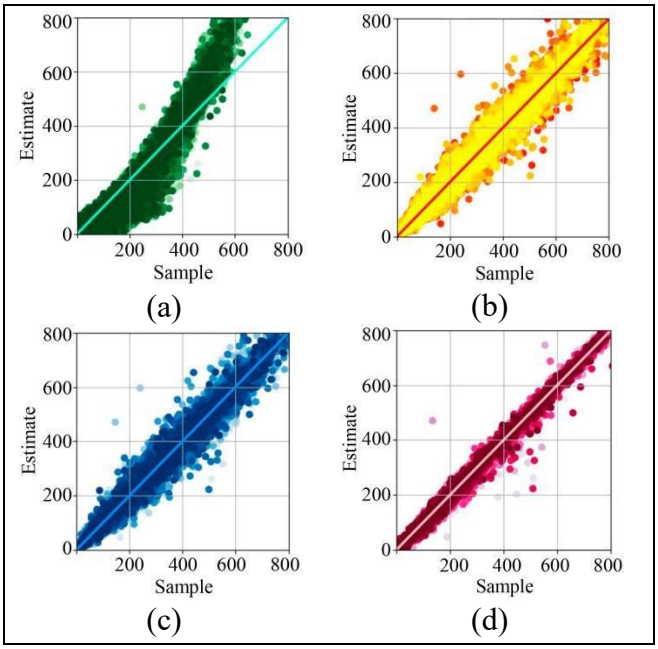

Fig. 7. The scatter diagrams of regression models: (a) LSM, (b) decision tree, (c) gradient boosting, and (d) artificial neural network.

Table 3

Values of the metrics for different forecasting algorithms (the end-to-end delay)

Model	Metrics		
	R	STD	R^2
LSM	0.926	66.31	0.85
Decision tree	0.990	24.48	0.98
Gradient boosting	0.990	24.71	0.98
Artificial neural network	0.998	12.23	0.99

Consider again the tandem queuing system with lengths $N = [1, 5, 10, 20]$ and estimate the probability of packet delivery for the artificial neural network. We selected a multilayer perceptron with 3 hidden layers, each containing 16 neurons (Fig. 8), and the sigmoidal activation function.

Logistic regression forecasted packet delivery with a large share of errors. The decision tree with a depth of 10 demonstrated better results. The best forecasts for the classification problem were obtained using the gradient boosting method and the artificial neural network. The values of the classification metrics are combined in Table 4.

Figure 9 shows the estimated probabilities of packet delivery in all models for the network size of $N = 10$. The trend indicates the actual values of the probability $1 - P_L$ for different loading coefficients ρ. The green color corresponds to the forecasts of packet delivery $(1 - P_L \geqslant B)$ and the red to those of packet loss $(1 - P_L < B)$. Among all the models, gradient boosting and neural network stand out as the ones providing the most accurate forecasts.

Table 4

Values of the metrics for different classification models (the probability of packet delivery)

Model	Metrics		
	Precision	Recall	F_1
Logistic regression	0.804	0.821	0.813
Decision tree	0.9618	0.905	0.912
Gradient boosting	0.966	0.969	0.968
Artificial neural network	0.977	0.951	0.964

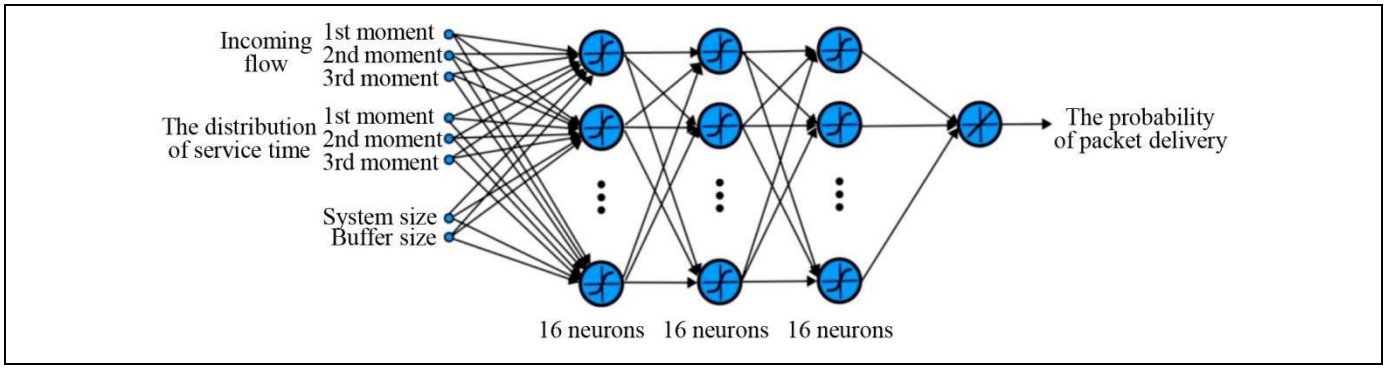

Fig. 8. The neural network architecture for classifying the tandem queuing system by the probability of packet delivery.

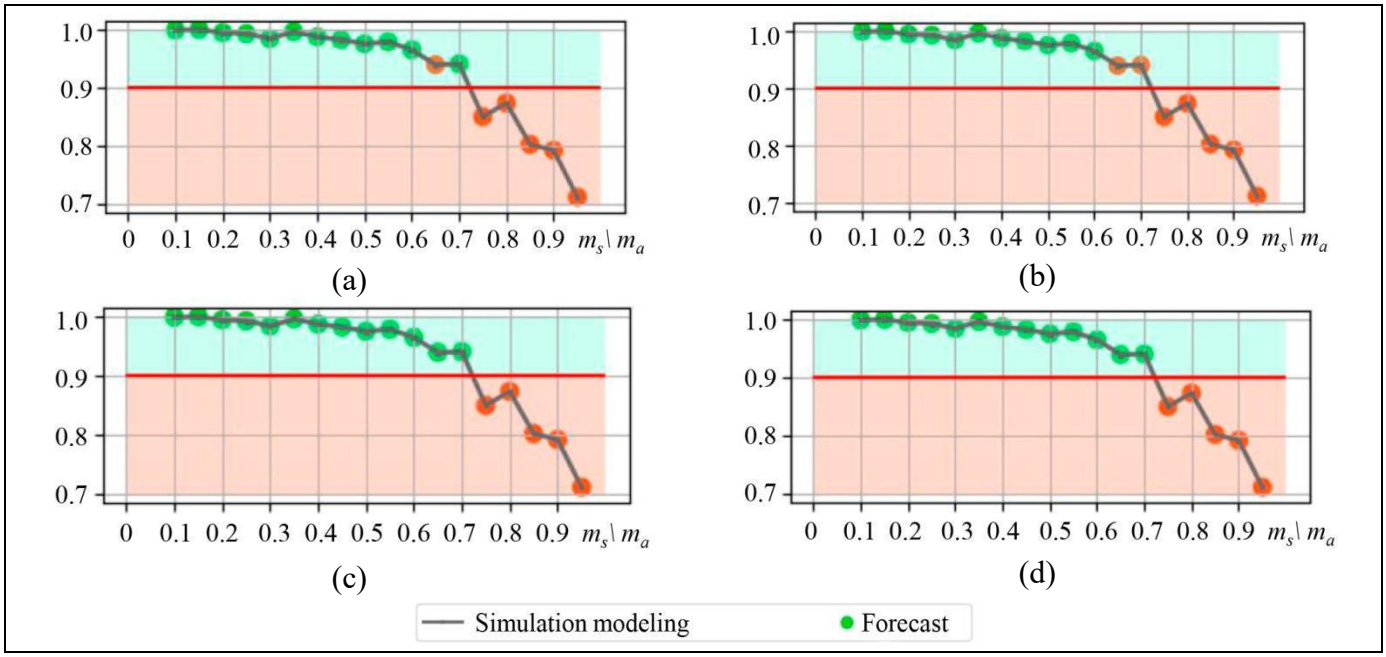

Fig. 9. The estimates of the probability of packet delivery for the tandem queuing system of size 10: (a) logistic regression, (b) decision tree, (c) gradient boosting, and (d) artificial neural network.

4.3. Calculation Time Analysis

For the obtained models, the calculation times of the steady-state performance characteristics by different methods are presented in Table 5. The calculation times of the simulation model depending on the system loading coefficient are also presented for comparison. The sample size is 360 rows. It is reasonable to compare the calculation time for the entire dataset because the time varies in individual cases depending on the tandem length. Note that the calculation time is the time to obtain the final result. Obviously, in each new model run, the time will also vary depending on the CPU occupancy of the PC. Also, the simulation model calculates two network performance characteristics per run. Machine learning models typically take much less time for calculations than simulation models.

Table 5

The time to calculate performance characteristics

Model	Calculation time, s
End-to-end delay	
Simulation model	172.2
LSM	$4.8 \cdot 10^{-6}$
Decision tree	$5.5 \cdot 10^{-6}$
Gradient boosting	$5 \cdot 10^{-6}$
Artificial neural network	$5.7 \cdot 10^{-6}$
The probability of packet delivery	
Simulation model	172.2
Logistic regression	$5.3 \cdot 10^{-6}$
Decision tree	$5 \cdot 10^{-6}$
Gradient boosting	$4.8 \cdot 10^{-6}$
Artificial neural network	$5.3 \cdot 10^{-6}$

CONCLUSIONS

In this paper, we have developed new approaches to studying the performance characteristics of tandem queuing systems. We have described a precise algorithm for calculating a low-dimensional tandem queuing system as well as its complexity estimates and application to the validation of a simulation model. For the investigation of high-dimensional tandem queuing systems, we have proposed an effective approach based on a combination of simulation modeling and machine learning methods. Several machine learning methods, including decision trees, gradient boosting, artificial neural networks, etc., have been comparatively analyzed. According to the numerical examples, machine learning methods demonstrate high effectiveness as well as a sharp reduction in the calculation time.

Acknowledgments. *This work was supported by the Russian Science Foundation, project no. 22-49-02023, https://rscf.ru/project/22-49-02023/.*

REFERENCES

1. Khayyati, S. and Tan, B., Supervised-Learning-Based Approximation Method for Multi-Server Queueing Networks under Different Service Disciplines with Correlated Interarrival and Service Times, *International Journal of Production Research*, 2022, vol. 60, no. 17, pp. 5176–5200. https://doi.org/10.1080/00207543.2021.1951448.
2. Kumar, B.K., Sankar, R., Krishnan, R.N., and Rukmani, R., Performance Analysis of Multi-processor Two-Stage Tandem Call Center Retrial Queues with Non-Reliable Processors, *Methodology and Computing in Applied Probability*, 2022, vol. 24, no. 1, pp. 95–142. https://doi.org/10.1007/s11009-020-09842-6.
3. Oblakova, A., Al Hanbali, A., Boucherie, R.J., van Ommeren, J.C., and Zijm, W.H., An Analytical Model for a Tandem of Two Traffic-Light Intersections under Semi-Actuated and Fixed Control, *Transportation Research Interdisciplinary Perspectives*, 2022, vol. 16, art. no. 100715. https://doi.org/10.1016/j.trip.2022.100715.
4. Rovetto, C., Cruz, E., Nuñez, I., Santana, K., Smolarz, A., Rangel, J., and Cano, E.E., Minimizing Intersection Waiting Time: Proposal of a Queue Network Model Using Kendall's Notation in Panama City, *Applied Sciences*, 2023, vol. 13, no. 18, art. no. 10030. https://doi.org/10.3390/app131810030.
5. Pershin, O.Y., Mukhtarov, A.A., Vishnevsky, V.M., and Larionov, A.A., Optimal Placement of Base Stations in Integrated Design of Wireless Networks, *Programming and Computer Software*, 2023, vol. 49, suppl. 2, pp. S82–S90. https://doi.org/10.1134/S0361768823100055.
6. Vishnevsky, V., Krishnamoorthy, A., Kozyrev, D., and Larionov, A., Review of Methodology and Design of Broadband Wireless Networks with Linear Topology, *Indian Journal of Pure and Applied Mathematics*, 2016, vol. 47, no. 2, pp. 329–342. https://doi.org/10.1007/s13226-016-0190-7.
7. Gnedenko, B.W. and König, D., *Handbuch der Bedienungstheorie II*, Berlin: De Gruyter, 1984. https://doi.org/10.1515/9783112614747.
8. Neuts, M.F., Two Queues in Series with a Finite, Intermediate Waitingroom, *Journal of Applied Probability*, 1968, vol. 5, no. 1, pp. 123–142. https://doi.org/10.2307/3212081.
9. Dieleman, N.A., Berkhout, J., and Heidergott, B., A Neural Network Approach to Performance Analysis of Tandem Lines: the Value of Analytical Knowledge, *Computers and Operations Research*, 2023, vol. 152, no. 3, art. no. 106124. DOI: https://doi.org/10.1016/j.cor.2022.106124.
10. Dudin, S.A., Dudin, A.N., Dudina, O.S., and Chakravarthy, S.R., Analysis of a Tandem Queuing System with Blocking and Group Service in the Second Node, *International Journal of Systems Science: Operations and Logistics*, 2023, vol. 10, no. 1, art. no. 2235270. DOI: https://doi.org/10.1080/23302674.2023.2235270.
11. Dudin, S.A., Dudina, O.S., and Dudin, A.N., Analysis of Tandem Queue with Multi-Server Stages and Group Service at the Second Stage, *Axioms*, 2024, vol. 13, no. 4, art. no. 214. DOI: https://doi.org/10.3390/axioms13040214.

12. Bocharov, P.P., Manzo, R., and Pechinkin, A.V., Analysis of a Two-Phase Queueing System with a Markov Arrival Process and Losses, *Journal of Mathematical Sciences*, 2005, vol. 131, no. 3, pp. 5606–5613. https://doi.org/10.1007/s10958-005-0432-4.
13. Kim, C.S., Klimenok, V., and Taramin, O., A Tandem Retrial Queueing System with Two Markovian Flows and Reservation of Channels, *Computers and Operations Research*, 2010, vol. 37, no. 7, pp. 1238–1246. https://doi.org/10.1016/j.cor.2009.03.030.
14. Kim, C., Klimenok, V.I., and Dudin, A.N., Priority Tandem Queueing System with Retrials and Reservation of Channels as a Model of Call Center, *Computers and Industrial Engineering*, 2016, vol. 96, pp. 61–71. https://doi.org/10.1016/j.cie.2016.03.012.
15. Klimenok, V., Breuer, L., Tsarenkov, G., and Dudin, A., The BMAP/G/1/–>·/PH/1/M Tandem Queue with Losses, *Performance Evaluation*, 2005, vol. 61, no. 1, pp. 17–40. https://doi.org/10.1016/j.peva.2004.09.001.
16. Lian, Z. and Liu, L., A Tandem Network with MAP Inputs, *Operations Research Letters*, 2008, vol. 36, no. 2, pp. 189–195. https://doi.org/10.1016/j.orl.2007.04.004.
17. Vishnevskii, V.M. and Dudin, A.N., Queueing Systems with Correlated Arrival Flows and Their Applications to Modeling Telecommunication Networks, *Automation and Remote Control*, 2017, vol. 78, no. 8, pp. 1361–1403. https://doi.org/10.1134/S000511791708001X.
18. Dudin, A.N., Klimenok, V.I., and Vishnevsky, V.M., *The Theory of Queuing Systems with Correlated Flows*, Cham: Springer, 2019.
19. Bruell, S.C., Balbo, G., and Afshari, P.V., Mean Value Analysis of Mixed, Multiple Class BCMP Networks with Load Dependent Service Stations, *Performance Evaluation*, 1984, vol. 4, no. 4, pp. 241–260. https://doi.org/10.1016/0166-5316(84)90010-5.
20. Vishnevsky, V., Klimenok, V., Sokolov, A., and Larionov, A., Performance Evaluation of the Priority Multi-Server System MMAP/PH/M/N Using Machine Learning Methods, *Mathematics*, 2021, vol. 9, no. 24, art. no. 3236. https://doi.org/10.3390/math9243236.
21. Klimenok, V., Dudin, A., and Vishnevsky, V., On the Stationary Distribution of Tandem Queue Consisting of a Finite Number of Stations, in *Communications in Computer and Information Science*, Springer, 2012, vol. 291, pp. 383–392. https://doi.org/10.1007/978-3-642-31217-5_40.
22. Palomo, S. and Pender, J., Learning the Tandem Network Lindley Recursion, *Proceedings of the 2021 Winter Simulation Conference (WSC)*, Phoenix, USA, December 2021, pp. 1–12. https://doi.org/10.1109/WSC52266.2021.9715530.
23. Rabta, B., A Review of Decomposition Methods for Open Queueing Networks, in *Rapid Modelling for Increasing Competitiveness*, Reiner, G., Ed., London: Springer, 2009, pp. 25–42. https://doi.org/10.1007/978-1-84882-748-6_3.
24. Vishnevsky, V., Larionov, A., Roman, I., and Semenova, O., Estimation of IEEE 802.11 DCF Access Performance in Wireless Networks with Linear Topology Using PH Service Time Approximations and MAP Input, *Proceedings of the 11th IEEE International Conference on Application of Information and Communication Technologies (AICT 2017)*, Moscow, 2017, pp. 1–5. https://doi.org/10.1109/ICAICT.2017.8687247.
25. Gorbunova, A.V., Vishnevsky, V.M., and Larionov, A.A., Evaluation of the End-to-End Delay of a Multiphase Queuing System Using Artificial Neural Networks, in *Lecture Notes in Computer Science*, Cham: Springer, 2021, vol. 12563, pp. 631–642. https://doi.org/10.1007/978-3-030-66471-8_48.
26. Kudou, T., Nii, S., and Okuda, T., A Performance Evaluation of Tandem Queueing Systems by Machine Learning, *Proceedings of the 2022 IEEE International Conference on Consumer Electronics (ICCE-Taiwan 2022)*, Taiwan, 2022, pp. 389–390. https://doi.org/10.1109/ICCE-Taiwan55306.2022.9869030.
27. Kudou, T. And Okuda, T., A Time Series Analysis of Single Server Queueing Systems by Using Machine Learning, *Proceedings of the 2023 IEEE International Conference on Consumer Electronics (ICCE-Taiwan 2023)*, Taiwan, 2023, pp. 327–328. https://doi.org/10.1109/ICCE-Taiwan58799.2023.10226861.
28. Vishnevsky, V.M., *Teoreticheskie osnovy proektirovaniya komp'yuternykh setei* (Theoretical Foundations of Computer Network Design), Moscow: Tekhnosfera, 2003. (In Russian.)
29. Vishnevsky, V.M., Klimenok, V.I., Sokolov, A.M., and Larionov, A.A., Investigation of the Fork–Join System with Markovian Arrival Process Arrivals and Phase-Type Service Time Distribution Using Machine Learning Methods, *Mathematics*, 2024, vol. 12, no. 5, art. no. 659. https://doi.org/10.3390/math12050659.
30. Efrosinin, D., Vishnevsky, V., and Stepanova, N., Optimal Scheduling in General Multi-Queue System by Combining Simulation and Neural Network Techniques, *Sensors*, 2023, vol. 23, no. 12, art. no. 5479. https://doi.org/10.3390/s23125479.
31. Telek, M. and Heindl, A., Matching Moments for Acyclic Discrete and Continuous Phase-Type Distributions of Second Order, *International Journal of Simulation Systems, Science and Technology*, 2002, vol. 3, no. 3, pp. 47–57.
32. Johnson, M.A. and Taaffe, M.R., Matching Moments to Phase Distributions: Mixtures of Erlang Distributions of Common Order, *Communications in Statistics. Stochastic Models*, 1989, vol. 5, no. 4, pp. 711–743. https://doi.org/10.1080/15326348908807131.
33. Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J., *Classification and Regression Trees*, New York: Chapman and Hall/CRC, 1984. https://doi.org/10.1201/9781315139470.
34. Demidova, L.A. and Usachev, P.O., Development and Approbation of the Improved CART Algorithm Version, *Journal of Physics: Conference Series*, 2020, vol. 1479, no. 1, art. no. 012085. https://doi.org/10.1088/1742-6596/1479/1/012085.
35. Gordon, A.D., A Review of *Classification and Regression Trees* by L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Biometrics*, 1984, vol. 40, no. 3, p. 874. https://doi.org/10.2307/2530946.
36. Loh, W.Y., Classification and Regression Trees, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2011, vol. 1, no. 1, pp. 14–23. https://doi.org/10.1002/widm.8.
37. Friedman, J.H., Stochastic Gradient Boosting, *Computational Statistics and Data Analysis*, 2002, vol. 38, no. 4, pp. 367–378. https://doi.org/10.1016/S0167-9473(01)00065-2.
38. Kingma, D.P. and Ba, J.L., Adam: A Method for Stochastic Optimization, *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*, http://arxiv.org/abs/1412.6980.

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

Received August 1, 2024,
and revised September 6, 2024.
Accepted September 13, 2024.

Author information

Vishnevsky, Vladimir Mironovich. Dr. Sci. (Eng.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ vishn@inbox.ru
ORCID iD: https://orcid.org/0000-0001-7373-4847

Larionov, Andrei Alekseevich. Cand. Sci. (Eng.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ larioandr@gmail.com
ORCID iD: https://orcid.org/0000−0003−0539−0442

Mukhtarov, Amir Amangel'dyevich. Cand. Sci. (Eng.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ mukhtarov.amir.a@gmail.com
ORCID iD: https://orcid.org/0000−0002−8191−6381

Sokolov, Aleksandr Mikhailovich. Researcher, Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ aleksandr.sokolov@phystech.edu
ORCID iD: https://orcid.org/0000−0002−3589−5700

Cite this paper

Vishnevsky, V.M., Larionov, A.A., Mukhtarov, A.A., and Sokolov, A.M., Investigation of Tandem Queuing Systems Using Machine Learning Methods. *Control Sciences* **4**, 10–21 (2024). http://doi.org/10.25728/cs.2024.4.2

Original Russian Text © Vishnevsky, V.M., Larionov, A.A., Mukhtarov, A.A., Sokolov, A.M., 2024, published in *Problemy Upravleniya*, 2024, no. 4, pp. 13–25.

Translated into English by the authors;
finally edited by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences,
Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com

DOI: http://doi.org/10.25728/cs.2024.4.3

DESIGN OF SELF-CHECKING DIGITAL DEVICES WITH BOOLEAN SIGNALS CORRECTION USING WEIGHT-BASED BOSE–LIN CODES

D. V. Efanov*,** and Y. I. Yelina*

*Peter the Great Saint Petersburg Polytechnic University, St. Petersburg, Russia
**Russian University of Transport, Moscow, Russia

*✉ TrES-4b@yandex.ru, **✉ eseniya-elina@mail.ru

Abstract. This paper proposes a method for designing self-checking digital devices with Boolean signals correction and weight-based Bose–Lin codes. Unlike previous studies, the method involves Boolean signals correction (BSC) in the concurrent error-detection (CED) circuit for those functions describing the outputs of source devices that participate in the formation of data symbols of weight-based Bose–Lin codes. In such codes, as in the absolute majority of uniform separable codes, several data vectors correspond to the same check vector; therefore, it is possible to choose a method for determining BSC functions. We describe an algorithm for determining their values for each input combination, considering the testability of the checker and transformation elements in the CED circuit. The method involves the so-called "base" structure for monitoring multi-output devices by output groups. With this method, the designer of a self-checking device has high variability in choosing the design method and can regulate important indicators (structure redundancy, controllability, energy consumption, and others). Experiments with combinational benchmarks from MCNC Benchmarks were carried out. According to the experimental data, the method has high efficiency in terms of structure redundancy compared to the duplication method widespread in practice. The method can be effective when designing real devices with fault detection used in all areas of technology, including critical application systems in industry and transport.

Keywords: self-checking device, concurrent error-detection circuit, Boolean signals correction, weight-based sum code, weight-based Bose–Lin code.

INTRODUCTION

When designing highly reliable and safe discrete blocks and nodes of critical application systems, it is important to ensure the timely detection of faults and computational errors arising during their operation [1–4]. This property is achieved by off-line and on-line testing equipment [5–7].

One approach to designing discrete systems with fault detection is the development of devices with testable and self-checking structures [8–12]. This requires the introduction of redundancy, according to certain principles, into a source device (further called an object under diagnosis) or the use of external technical diagnosis means, including the equipment of objects under diagnosis with self-checking concurrent error-detection (CED) circuits [1, 13].

CED circuits are designed so that it is possible to judge the correct operation of an object under diagnosis by the values of the functions computed by this object or the functions formed in its structure at special checkpoints. Physical defects lead to the occurrence of faults at the outputs of the internal structure elements of an object under diagnosis and, as a consequence, to the appearance of distortions in the values of computed functions, which is detected by a CED circuit. For example, diagnostic attributes are the belonging of binary vectors formed in a CED circuit to a set of codewords of any predetermined redundant binary codes [1] or the belonging of functions formed in

a CED circuit to a predetermined special class of Boolean functions (e.g., linear, monotonic, or self-dual) [14, 15]. Several diagnostic features can be used together and simultaneously [16, 17].

There exist two main approaches to organizing CED circuits for discrete devices. The first (classical) one consists in that an object under diagnosis is equipped with a CED circuit in which the data vector formed at the object's outputs is supplemented with the check vector computed using a check logic block [1]. A checker is installed to verify the mutual correspondence of the data and check vectors. The second approach implies correction of the functions computed by the object under diagnosis in the CED circuit so that the codeword of a preselected redundant code is formed or the functions turn out to belong to special classes of Boolean functions [11]. This approach is based on Boolean signals correction (BSC) in a CED circuit. Being less investigated, it allows designing much more variants of CED circuits for the same object under diagnosis than the first approach [18]. As a result, the design problem of a self-checking device becomes simpler; moreover, the designer can tune the indicators of structure redundancy, controllability, energy consumption, etc. Using BSC makes it possible to design a self-checking device even when other methods become inapplicable, e.g., the duplication method widespread in practice [19].

This paper presents new results in the field of application of BSC together with binary redundant codes. We propose a design method for self-checking combinational devices with BSC and weight-based sum codes in the ring of modulo M residues (weight-based Bose–Lin codes).

1. THE STRUCTURE OF A CED CIRCUIT WITH BOOLEAN SIGNALS CORRECTION

Figure 1 shows the structural diagram of a CED circuit based on BSC. Here, the object under diagnosis is a block $F(X)$, equipped with t inputs and n outputs. Boolean vectors $\langle X\rangle = \langle x_t\, x_{t-1} \ldots x_2\, x_1\rangle$ are supplied at the inputs of this block during its operation. They are to compute the values of functions implemented by the block $F(X)$ and to form a Boolean vector $\langle f_n(X)\, f_{n-1}(X)\, \ldots\, f_2(X)\, f_1(X)\rangle$. The faults occurring during the operation of the block $F(X)$ distort the values of the vector $\langle f_n(X)\, f_{n-1}(X)\, \ldots\, f_2(X)\, f_1(X)\rangle$ formed at its outputs. By monitoring the occurrence of these distortions, it is possible to indirectly determine the presence of faults in the object under diagnosis [1].

A CED circuit is installed to check computations at the outputs of the object of diagnosis. In contrast to the classical structure (e.g., see [1]), the CED circuit in Fig. 1 performs not the complement of the vector $\langle f_n(X)\, f_{n-1}(X)\, \ldots\, f_2(X)\, f_1(X)\rangle$ but Boolean signals correction using modulo $M = 2$ addition (XOR gates). The vector $\langle f_n(X)\, f_{n-1}(X)\, \ldots\, f_2(X)\, f_1(X)\rangle$ is transformed into a vector $\langle h_n(X)\, h_{n-1}(X)\, \ldots\, h_2(X)\, h_1(X)\rangle$, which may have special diagnostic properties: for example, it may belong to the set of codewords of a preselected uniform binary code.

Each value $f_i(X), i = \overline{1, n},$ is corrected using two-input XOR gates: the first inputs receive the values $f_i(X)$ and the second ones the values of the same-name correction functions $g_i(X)$ of the same name. They are computed for the same sets of input variables as the values $f_i(X)$ by the correction function block $G(X)$.

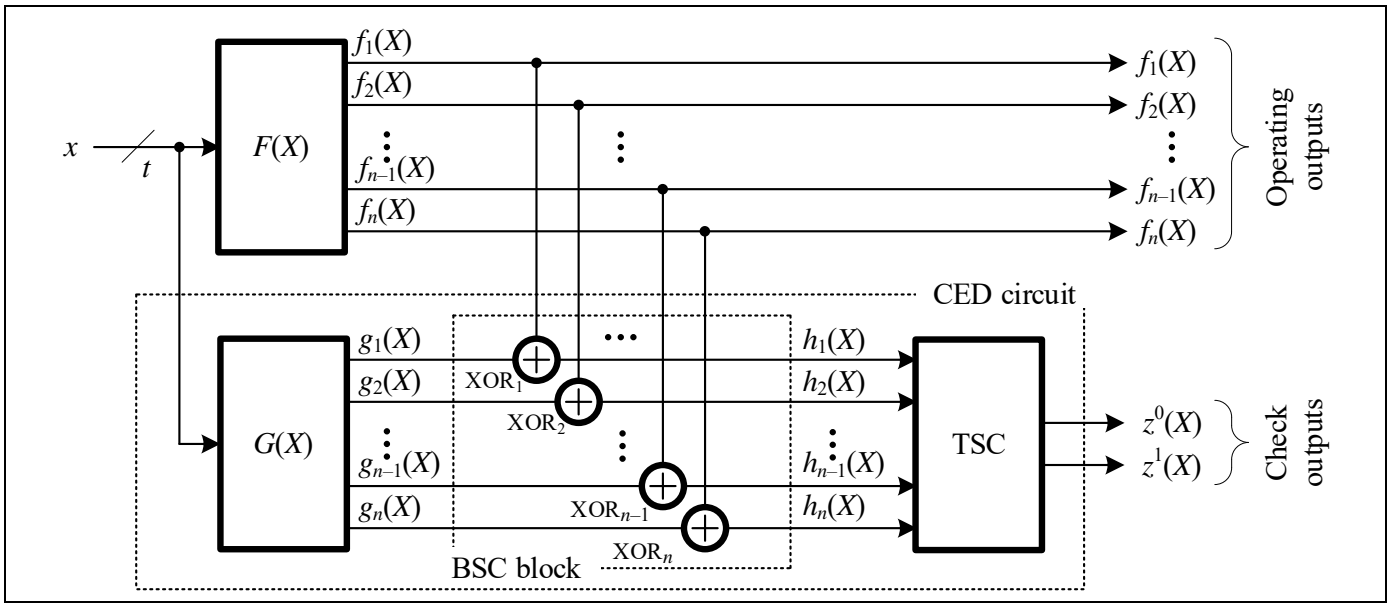

Fig. 1. The structural diagram of a CED circuit based on Boolean signals correction.

Thus, the transformations in the CED circuit are performed by the rule

$$h_i(X) = f_i(X) \oplus g_i(X), i = \overline{1, n}.$$

The cascade of XOR gates forms the Boolean signals correction block (the BSC block). A totally self-checking checker (TSC) is installed to verify whether the vector $<h_n(X)\ h_{n-1}(X)\ \dots\ h_2(X)\ h_1(X)>$ formed at the BSC block outputs belongs to the set of codewords of the code selected for checking computations. This device has two outputs $z^0(X)$ and $z^1(X)$, which are the check outputs of the CED circuit: the presence of a two-rail signal <01> or <10> at the outputs indicates of no computational errors at the outputs of the object under diagnosis and in the CED circuit blocks; violation of the two-rail property points to the presence of errors.

The structure in Fig. 1 has an important peculiarity that many CED circuits can be built for one selected uniform binary code. This peculiarity allows the designer to solve the most difficult task of organizing a CED circuit, i.e., to ensure its self-checking structure. In addition, the designer can regulate the indicators of structure redundancy of the self-checking device. With a preselected code, an available TSC structure, and a given implementation of the block $G(X)$, the self-checking device based on the classical CED circuit [1] has a single implementation, which complicates the total self-checking of the CED circuit and, in some cases, makes it impossible. For the case of a Berger code and a repetition code underlying a redundancy system, an example demonstrating this feature of the classical CED circuit structure was provided in [18]. Thus, even duplication with comparison of computational results, a widespread method in practice, does not always yield a self-checking CED circuit due to the difficulties in ensuring the comparator's self-checking. With the CED circuit structure based on BSC, the designer can build self-checking devices even when other methods turn out inapplicable.

The key to organizing a CED circuit based on BSC is the choice of a diagnostic attribute. As noted above, it can be the belonging of the vector $<h_n(X)\ h_{n-1}(X)\ \dots\ h_2(X)\ h_1(X)>$ formed at the BSC block outputs to a given uniform binary code. Then the properties of error detection in the codewords of this code will determine the possibilities of detecting distortions at the outputs of the object under diagnosis. Naturally, different uniform binary codes have different error detection properties by their types (by the number of combinations of distortions of zero and unit values) and multiplicities [20, 21]. The choice of an appropriate code for checking computations becomes determinant from the viewpoints of covering the errors occurring at the outputs of the object of diagnosis and ensuring the self-checking of the CED circuit.

The following conditions must be satisfied for a CED circuit to be self-checking. During the operation of a self-checking device, it is necessary to supply a given set of input signals and ensure:

- the checkability of the block $G(X)$, i.e., the possibility of manifesting any fault from a given model as distortions at its outputs for at least one set $<x_t\ x_{t-1}\ \dots\ x_2\ x_1>$ [5];
- formation of check tests for transformation elements in the BSC block. The full test of the XOR gate in its canonical implementation contains all four combinations {00, 01, 10, 11} [22];
- formation of the check test for the TSC, which is related to the redundant code chosen at the design stage and the implementation peculiarities of the TSC structure [23].

Any uniform binary block codes (both non-separable and separable) can be chosen as base codes for designing the CED circuit shown in Fig. 1. In the case of non-separable codes, it is impossible to distinguish data and check symbols in codewords; in the case of separable codes, data symbols can be combined into the data vector and check symbols into the check vector. When choosing an appropriate code, it is reasonable to start from some threshold: the cardinality of the set of codewords of a given non-separable code or the number of check symbols of a given separable code. This threshold should not exceed the value $k = n$, which is the number of check symbols in codes with repetition, underlying the duplication structure widespread in practice. Duplication provides coverage of any error combinations at the outputs of the object under diagnosis but, at the same time, leads to significant structure redundancy: the implementation complexity indicators of a device with a CED circuit can be 3–4 (and even more) times higher than those of the object under diagnosis. For this reason, codes with low redundancy are often considered when designing CED circuits. Such codes include non-separable constant-weight codes and separable sum codes [1, 2, 11, 20, 21, 23]. Special attention is paid to the codes with redundancy close to the minimum possible for detecting errors at the outputs of objects under diagnosis. The lower bound $k = 1$ is achieved when using a parity code. Therefore, it is interesting to study the applicability of redundant parity codes with $k = 2$ (and a close number of check symbols) to checking computations.

To construct a code with $k = 2$ check symbols, we can count the number of significant signals in the data vector in the ring of modulo $M = 4$ residues. Then the so-called residue or modular sum codes will be constructed [1]. They are often termed Bose–Lin codes in

the literature [23]. For example, when designing CED circuits based on the classical structure with the complement of the data vector formed at the outputs of the object under diagnosis by the check vector, the advantages of using such codes were demonstrated in [24], including the gain compared to duplication and a Berger code applied for checking computations. Bose–Lin codes can also be used to design CED circuits based on BSC. A CED circuit design method based on BSC with such codes was presented in [25]; this method involves the transformation of only some values of the vector $<f_n(X)\, f_{n-1}(X)\, \ldots\, f_2(X)\, f_1(X)>$, responsible for forming check symbols of Bose–Lin codes.

Despite their advantages of simple construction, Bose–Lin codes are not optimal from the viewpoint of error detection in data vectors with given m and k (the numbers of data and check symbols, respectively). Such are the codes where all 2^m data vectors can be uniformly distributed into groups corresponding to all 2^k check vectors [26]. There is a method for constructing some modified Bose–Lin code with modulus $M = 4$ possessing this property.

The problem of covering errors at the outputs of an object under diagnosis using uniform redundant binary codes was considered in many studies. Its solution for different codes was described, e.g., in [1, 20, 21]. Let us dwell on the solution of the most difficult problem of ensuring the self-checking of CED circuits.

2. THE "BASE" CODE FOR ORGANIZING THE CED CIRCUIT STRUCTURE

Let us modify the Bose–Lin code as follows. Fixing the value m (the number of data symbols), we form a data vector $<y_m\, y_{m-1}\, \ldots\, y_2\, y_1>$ and assign to this data vector an array of weights $[w_m, w_{m-1}, \ldots, w_2, w_1]$, $w_i \in \mathbb{N}$. Next, we determine the modulo $M = 4$ residue of the total weight of significant bits instead of the number of significant bits (the weight of the data vector) in the ring of modulo $M = 4$ residues:

$$W_4 = \sum_{i=1}^{m} y_i w_i \,(\mathrm{mod}\, 4).$$

For each data vector, the binary analog of the number W_4 is written into the bits of the check vector.

Let $WS(m, k, M)$, where $k = 2$ and $M = 4$, i.e., $WS(m, 2, 4)$, denote the modified Bose–Lin code. Consider such a code for $m = 4$.

Different combinations of weights yield a large number of $WS(4, 2, 4)$-codes. In this case, each weight can be only a number from the set {1, 2, 3} since the value of the smallest non-negative residue is written into the check vector in binary form. When setting the weights $w_i = 4j(\mathrm{mod}\ 4) = 0$, $j \in \mathbb{N}_0$, a code susceptible to interference will be constructed. (The bits whose weight is a multiple of the modulus will not be checked.)

For $WS(4, 2, 4)$-codes, there are 15 ways of constructing an interference-immune code, defined by the ways of weighting the data symbols $[w_4, w_3, w_2, w_1]$: [1, 1, 1, 1], [1, 1, 1, 2], [1, 1, 1, 3], [1, 1, 2, 2], [1, 1, 2, 3], [1, 1, 3, 3], [1, 2, 2, 2], [1, 2, 2, 3], [1, 2, 3, 3], [1, 3, 3, 3], [2, 2, 2, 2], [2, 2, 2, 3], [2, 2, 3, 3], [2, 3, 3, 3], [3, 3, 3, 3]. Note that possible permutations of weights in the arrays that do not affect the general properties of error detection in codewords are neglected here.

All $WS(4, 2, 4)$-codes with odd numbers as weights do not detect 54 errors in data vectors; $WS(4, 2, 4)$-codes with all even weights ([2, 2, 2, 2]) do not detect 112 errors in data vectors; other $WS(4, 2, 4)$-codes do not detect 48 errors in data vectors, the minimum possible number under given M and m. In all symbols of codewords, all $WS(4, 2, 4)$-codes do not detect 240 errors.

From the viewpoints of error detection in data vectors and the self-checking of test equipment for a selected separable code, it is important how many data vectors correspond to the same check vector. All data vectors can be classified into groups corresponding to the same check vector. The maximum of 2^k check vectors can be generated for a code with k check symbols. If all 2^m data vectors of the code are uniformly distributed between 2^k check vectors, such a code will detect the maximum number of errors in the data vectors. In addition, the self-checking of test equipment will be ensured much easier. For example, checkers of separable codes are most simply built in the form of two-cascade structures containing an encoder and a comparator [27]. To fully test them, it will be necessary to generate, at least once, each check vector at the comparator inputs. This is impossible if the number of check vectors of the code is not maximal and not equal to 2^k. For example, the classical Berger codes widely known have the maximum number of check vectors for k check symbols only in the case $m = 2^k - 1$. For $m \neq 2^k - 1$, the full set of check vectors is not formed for Berger codes [28]. Implementing a totally self-checking CED circuit becomes difficult. For the modified Bose–Lin codes considered in this paper, all possible check vectors are formed for k check symbols.

Consider the use of a $WS(4, 2, 4)$-code with the weights [2, 2, 2, 3] as an example. Table 1 classifies the data vectors into groups corresponding to identical check vectors for the selected $WS(4, 2, 4)$-code. According to the table, this code has the minimum possible number of undetectable errors in data vectors, and each check vector corresponds to four data vectors.

Table 1

Classification of data vectors by identical check vectors

W_4			
0	1	2	3
Check vectors			
00	01	10	11
Data vectors			
0000	0011	0010	0001
0110	0101	0100	0111
1010	1001	1000	1011
1100	1111	1110	1101

This simplifies the task of ensuring encoder testability in the CED circuit for real devices, where (as a rule) data vectors at the outputs are formed unevenly.

When constructing a *WS*(4, 2, 4)-code, one may assign other weights to obtain a different classification table of data vectors by identical check vectors. This task is not studied here. We emphasize that the encoder of the *WS*(4, 2, 4)-code under consideration, which is suitable for designing CED circuits based on BSC and contains a larger number of even weights, will have one of the simplest structures among the encoders of codes with other weights; in addition, the code will not detect the minimum possible total number of errors in data vectors.

3. THE "BASE" STRUCTURE FOR CED CIRCUIT DESIGN

We demonstrate the application of the Bose–Lin code for the structure in Fig. 1.

Figure 2 shows the "base" structure for CED circuit design for a group of six outputs of an object under diagnosis based on the *WS*(4, 2, 4)-code. Here, the values of only two functions implemented by the object are corrected in the CED circuit to form the check symbols of the code. This structure was investigated earlier, e.g., in [29], but in the case of the weights [4, 3, 2, 1]. For these weights, self-checking CED circuits can be designed only in some special cases: since the most significant bit is not controlled by check symbols, the formation of some data vectors should be excluded.

In the structure shown in Fig. 2, a *WS*(4, 2, 4)-code is used to organize checking. In the CED circuit, the codewords $<h_6(X)\ h_5(X)\ h_4(X)\ h_3(X)\ h_2(X)\ h_1(X)>$ of this code are formed at the BSC block outputs (i.e., at the TSC inputs). The two lower bits correspond to check symbols and the four higher bits to data symbols. Data symbols are formed directly at the outputs $f_3(X)$, $f_4(X)$, $f_5(X)$, and $f_6(X)$ of the object under diagnosis. The check symbols are calculated using the formulas $h_1(X) = f_1(X) \oplus g_1(X)$ and $h_2(X) = f_2(X) \oplus g_2(X)$. The functions g_1 and g_2 represent the correction functions formed by the block $G(X)$. Thus, when supplying the sets $<x_t\ x_{t-1}\ \ldots\ x_2\ x_1>$ to the inputs in the CED circuit, any vector

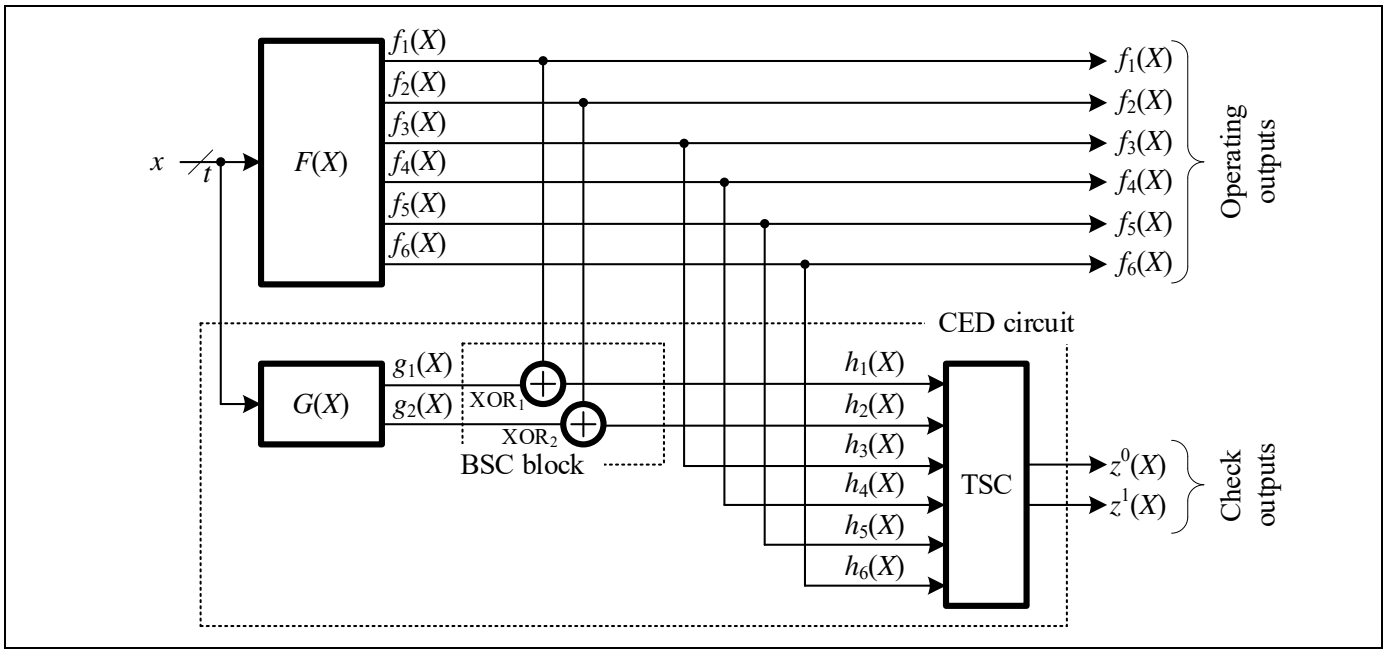

Fig. 2. The structural diagram of a CED circuit with transformation of the values of some functions implemented at the outputs of an object under diagnosis into the functions forming check symbols of the *WS*(4, 2, 4)-code.

$<f_6(X)\, f_5(X)\, f_4(X)\, f_3(X)\, f_2(X)\, f_1(X)>$ is transformed into the codeword $<h_6(X)\, h_5(X)\, h_4(X)\, h_3(X)\, h_2(X)\, h_1(X)>$, which belongs to the *WS*(4, 2, 4)-code. The checker of this code has a typical structure containing an encoder and a two-rail checker (TRC) [30]. Note that the TRC block operates in the two-rail logic; therefore, it is required to pre-invert the signals from the output of the encoder of the *WS*(4, 2, 4)-code or invert the signals $h_1(X)$ and $h_2(X)$.

Let us slightly modify the structure in Fig. 2. Consider the functions implemented by the object under diagnosis and correct the values of those forming the data symbols of the *WS*(4, 2, 4)-code (Fig. 3). In this structure, only the check symbols of codewords are directly formed, and the data symbols are calculated using the block $G(X)$. Otherwise, this structure is similar to the one in Fig. 2. However, when building this structure, it is possible to ensure the self-checking of the end device by selecting the data vectors corresponding to the check vectors (see Table 1), particularly in those cases where the structure in Fig. 2 fails with this task.

The peculiarity of the modified structure is that the check vectors of the *WS*(4, 2, 4)-code in the CED circuit will be formed unambiguously, but the corresponding data vectors can be selected from four variants for each check vector (see Table 1). This peculiarity provides high variability when building the "base" structure and allows obtaining several variants of the block $G(X)$. In turn, it becomes possible to form a complete set of test combinations for XOR gates and regulate the structure redundancy indicators of the CED circuit.

Let us determine the number of ways to build the CED circuit according to the structure in Fig. 3. There are $C_{4+2}^4 = 15$ variants for selecting the transformed outputs. There are $P_4 = 4! = 24$ variants for placing data symbols in the data vector. There are $P_2 = 2! = 2$ variants for placing check symbols in the check vector. The product of the three values mentioned characterizes the number of ways to choose the sequences of data and check symbols. Besides, on each input combination, there are exactly four ways to determine the four components of one data vector in the CED circuit: due to only 2^t input combinations, there are $4 \cdot 2^t$ ways of this determination process. The number of ways to build the CED circuit according to the "base" structure in Fig. 3 is given by

$$N_{WS(4,2,4),t} = 4 \cdot 2^t \cdot 15 \cdot 24 \cdot 2 = 2\,880 \cdot 2^t. \quad (1)$$

For example, for $t = 4$ inputs, we have $N_{WS(4,2,4),4} = 2\,880 \cdot 2^4 = 46\,080$.

In the case $t = 4$, applying the same code *WS*(4, 2, 4) to the structure in Fig. 2 yields $N_{WS(4,2,4),4} = 15 \cdot 24 \cdot 2 = 720$ ways to organize the CED circuit. (It is impossible to vary the values of data symbols.)

This number is 64 times smaller than that for the structure in Fig. 3. Hence, the CED circuit structure based on BSC with the *WS*(4, 2, 4)-code proposed

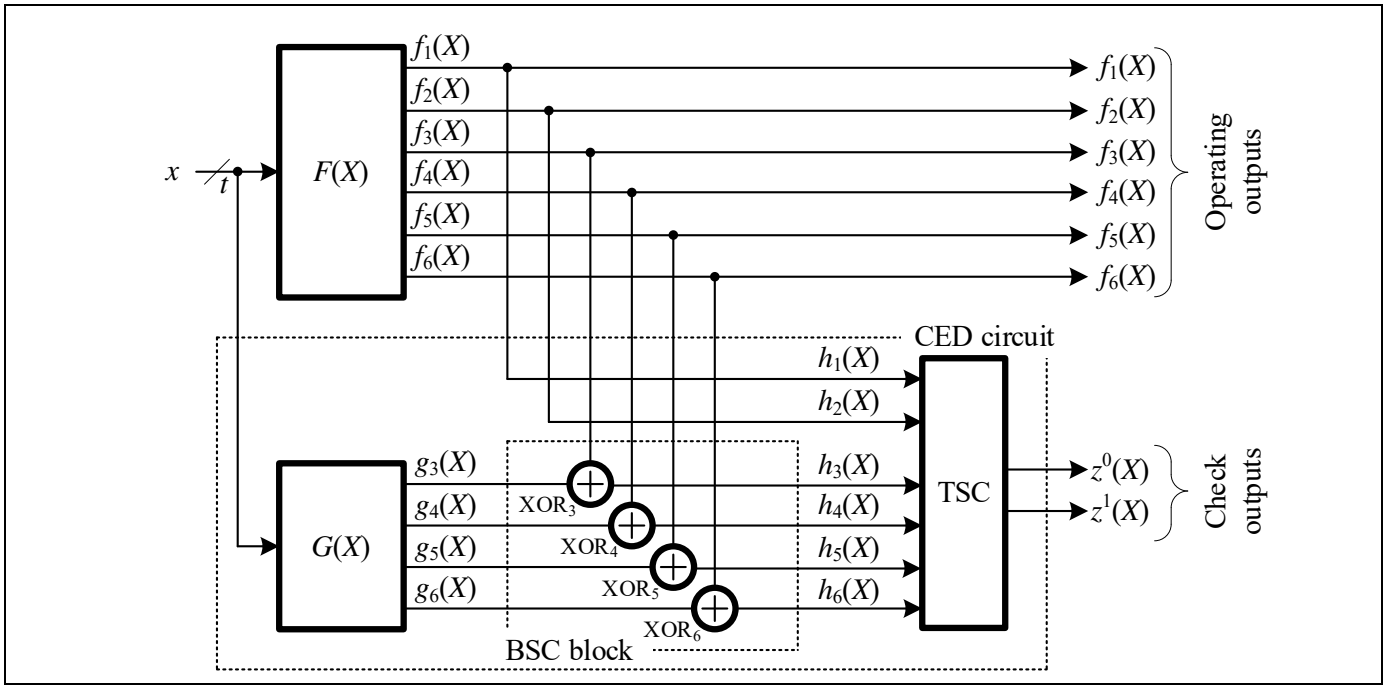

Fig. 3. The structural diagram of a CED circuit with transformation of the values of some functions implemented at the outputs of an object under diagnosis that form data symbols of the *WS*(4, 2, 4)-code.

in this paper has another advantage over the counterpart in Fig. 2, i.e., much higher variability in construction.

The "base" CED circuit structure is implemented for a six-output object under diagnosis. If an object under diagnosis has more outputs, they are divided into groups of six outputs each; for each group, a separate CED circuit is built according to the "base" structure with the subsequent union of the check outputs at the inputs of the self-checking comparator.

4. ALGORITHM FOR CED CIRCUIT DESIGN

At the design stage, it is required to ensure self-checking conditions of the CED circuit. For this purpose, the block $G(X)$ should be made checkable, which is related only to design methods: for this block, it is required to obtain a controllable structure with respect to a selected fault model that will manifest faults in the form of errors at its outputs when at least one set $<x_t\ x_{t-1}\ ...\ x_2\ x_1>$ is supplied to the inputs. It is also required to supply check tests to each element of the BSC block and the TSC. For each element of the BSC block, it is required to generate each combination from the set {00, 01, 10, 11} at least once during the operation of the self-checking device [22]. To test the checker, it is required to generate each check vector of the $WS(4, 2, 4)$-code at least once [20, 21]. We will implement a design algorithm for the CED circuit to ensure these conditions during the operation of the self-checking device.

The idea of the algorithm is as follows. When designing the CED circuit, it is required to unambiguously determine which codewords $<h_6(X)\ h_5(X)\ h_4(X)\ h_3(X)\ h_2(X)\ h_1(X)>$ of the $WS(4, 2, 4)$-code will be generated at the BSC block outputs for each set $<x_t\ x_{t-1}\ ...\ x_2\ x_1>$ arriving at the inputs. In the structure in Fig. 3, two symbols are unambiguously determined by the values $h_1(X) = f_1(X)$ and $h_2(X) = f_2(X)$, while those of the symbols $h_3(X)$, $h_4(X)$, $h_5(X)$, and $h_6(X)$ are considered indeterminate. According to Table 1, four data vectors will correspond to each check vector. Thus, at the CED circuit design stage, it is required to fix one of the four codewords of the $WS(4, 2, 4)$-code for each input set $<x_t\ x_{t-1}\ ...\ x_2\ x_1>$. This can be done arbitrarily. When designing the CED circuit, the main task is to determine the values of $h_3(X)$, $h_4(X)$, $h_5(X)$, and $h_6(X)$ at the BSC block outputs and then obtain the values of the correction functions computed by the block $G(X)$. The values of the correction functions are obtained for each input set $<x_t\ x_{t-1}\ ...\ x_2\ x_1>$ unambiguously because

$$h_i(X) = f_i(X) \oplus g_i(X) \Rightarrow g_i(X) = f_i(X) \oplus h_i(X).$$

The algorithm is intended to form the values of data symbols $h_3(X)$, $h_4(X)$, $h_5(X)$, and $h_6(X)$ considering the conditions of forming check tests for the elements of the BSC block and TSC.

According to the aforesaid, there are various algorithms for determining the values of the data symbols $h_3(X)$, $h_4(X)$, $h_5(X)$, and $h_6(X)$ at the BSC block outputs in the structure in Fig. 3 for each input set $<x_t\ x_{t-1}\ ...\ x_2\ x_1>$. Consider one of them.

A CED circuit design algorithm based on BSC using the $WS(4, 2, 4)$-code for a six-output device includes the following steps.

Step 1. For each function implemented at the outputs of the device $F(X)$, check whether it takes value 0 (and 1) for at least two sets of argument values. This is necessary to generate check tests for the XOR gates when determining the values of the data symbols $h_3(X)$, $h_4(X)$, $h_5(X)$, and $h_6(X)$ at the BSC block outputs.

Step 2. Select the outputs of the device $F(X)$ that are directly connected to the TSC and correspond to the check symbols of the $WS(4, 2, 4)$-code, e.g., the outputs $f_1(X)$ and $f_2(X)$. For these outputs, check the formation of each combination of values {00, 01, 10, 11} for at least one input set. This is necessary to form a check test for the TSC. If this condition fails, then other non-transformable outputs should be selected. If such a combination of two outputs does not exist, it will be impossible to build a self-checking CED circuit by the method under consideration.

Step 3. Form a value table for the outputs of the blocks $F(X)$, $G(X)$ and the BSC block, representing a truth table. This table should be filled in at the CED circuit design stage. It uniquely determines the values of $f_i(X)$ and $h_1(X) = f_1(X)$, $h_2(X) = f_2(X)$ for each input set $<x_t\ x_{t-1}\ ...\ x_2\ x_1>$. The values of the functions $h_3(X)$, $h_4(X)$, $h_5(X)$, and $h_6(X)$ and $g_3(X)$, $g_4(X)$, $g_5(X)$, and $g_6(X)$ are first considered indeterminate; they have to be obtained during the determination process.

Step 4. Fill the columns $h_3(X)$, $h_4(X)$, $h_5(X)$, and $h_6(X)$ row by row, starting from the column corresponding to the most significant bit of the data vector. Table 1 can be used for this purpose. For each of the functions f_5 and f_6, check whether it takes value 0 (and 1) for at least one set of the argument values from the first half (No. 0 ... $2^{t-1} - 1$), as well as for input sets from the second half (No. 2^{t-1} ... $2^t - 1$). Such a check excludes the subsequent check of the formation of all combinations from the check test for the transformation elements of the values of the functions f_5 and f_6 in the BSC block.

Step 5. For the first (second) half of the sets of argument values, assign values 0 (1, respectively) to the functions $h_6(X)$ and $h_5(X)$. As a result, the test combi-

nations <00> and <11> will be formed at least once for the first half of the input sets, and the test combinations <01> and <10> will be formed for the second half of the inputs sets for the XOR gates performing the transformations $h_6(X) = f_6(X) \oplus g_6(X)$ and $h_5(X) = f_5(X) \oplus g_5(X)$.

Step 6. Fill unambiguously the columns $h_3(X)$ and $h_4(X)$ in accordance with Table 1. Filling is unambiguous since the values of the higher bits of data vectors are determined.

Step 7. Check the condition of forming check tests for the elements performing the transformations $h_3(X) = f_3(X) \oplus g_3(X)$ and $h_4(X) = f_4(X) \oplus g_4(X)$. If the conditions are satisfied, proceed to Step 9; otherwise, change the way of filling the columns $h_5(X)$ and $h_6(X)$ by rearranging the outputs of the object under diagnosis.

Step 8. Determine the values of the functions $g_i(X) = f_i(X) \oplus h_i(X), i \in \{3, 4, 5, 6\}$.

Step 9. Optimize the functions using any known method [31].

Step 10. Design the block $G(X)$ in the selected components.

We demonstrate this algorithm on the following example.

Example. Let a combinational device be described in the first seven rows of Table 2. Here the sets of input arguments are numbered with decimals corresponding to the binary numbers formed by each set of input arguments.

At *Step 1*, we check that each function implemented at the outputs of the device $F(X)$ takes value 0 (and 1) for at least two sets of argument values. In the example under consideration, this condition is satisfied, which follows from the analysis of Table 2. Next, at *Step 2*, we determine the outputs of the object under diagnosis that are directly connected to the TSC. Assume that these are the outputs $f_1(X)$ and $f_2(X)$.

Step 3 consists in forming a value table for the outputs of the blocks $F(X)$, $G(X)$ and the BSC block (Table 2). At this step, we also unambiguously fill the values of the functions $f_i(X)$ and $h_1(X) = f_1(X)$, $h_2(X) = f_2(X)$ for each input set $<x_t\ x_{t-1}\ \ldots\ x_2\ x_1>$. The values of the functions $h_4(X)$ and $h_3(X)$ are indeterminate at this step. At *Step 4*, we check the formation of values 0 (and 1) of the functions f_5 and f_6 for at least one of the first and second halves of the sets of argument values. We fill the columns $h_6(X)$ and $h_5(X)$ at *Step 5* (Table 2). At *Step 6*, we unambiguously fill the columns $h_3(X)$ and $h_4(X)$. The filled table is presented below (see Table 3).

Table 2

Signals received when forming the values of functions h_6 and h_5

No.	$f_6(X)$	$f_5(X)$	$f_4(X)$	$f_3(X)$	$f_2(X)$	$f_1(X)$	$h_6(X)$	$h_5(X)$	$h_4(X)$	$h_3(X)$	$h_2(X)$	$h_1(X)$	$g_6(X)$	$g_5(X)$	$g_4(X)$	$g_3(X)$	Combinations at the inputs of XOR gates			
																	XOR_6	XOR_5	XOR_4	XOR_3
0	1	0	1	1	1	0	0	0	–	–	1	0	–	–	–	–	11	00	–	–
1	0	1	1	0	0	0	0	0	–	–	0	0	–	–	–	–	00	11	–	–
2	0	0	0	1	0	1	0	0	–	–	0	1	–	–	–	–	00	00	–	–
3	0	1	1	0	1	0	0	0	–	–	1	0	–	–	–	–	00	11	–	–
4	0	1	0	0	0	1	0	0	–	–	0	1	–	–	–	–	00	11	–	–
5	1	1	1	0	1	0	0	0	–	–	1	0	–	–	–	–	11	11	–	–
6	1	1	0	1	0	1	0	0	–	–	0	1	–	–	–	–	11	11	–	–
7	1	0	1	0	1	1	0	0	–	–	1	1	–	–	–	–	11	00	–	–
8	1	1	1	0	1	0	1	1	–	–	1	0	–	–	–	–	10	10	–	–
9	0	0	1	0	0	1	1	1	–	–	0	1	–	–	–	–	01	01	–	–
10	1	0	0	1	1	0	1	1	–	–	1	0	–	–	–	–	10	01	–	–
11	0	1	0	1	0	0	1	1	–	–	0	0	–	–	–	–	01	10	–	–
12	0	1	0	0	0	0	1	1	–	–	0	0	–	–	–	–	01	10	–	–
13	0	0	1	1	0	0	1	1	–	–	0	0	–	–	–	–	01	01	–	–
14	1	1	0	1	0	1	1	1	–	–	0	1	–	–	–	–	10	10	–	–
15	1	0	1	1	1	0	1	1	–	–	1	0	–	–	–	–	10	01	–	–

Note. The symbol "–" indicates the columns with indeterminate values before Step 7.

Table 3

The values of signals in the CED circuit

No.	$f_6(X)$	$f_5(X)$	$f_4(X)$	$f_3(X)$	$f_2(X)$	$f_1(X)$	$h_6(X)$	$h_5(X)$	$h_4(X)$	$h_3(X)$	$h_2(X)$	$h_1(X)$	$g_6(X)$	$g_5(X)$	$g_4(X)$	$g_3(X)$	Combinations at the inputs of XOR gates			
																	XOR_6	XOR_5	XOR_4	XOR_3
0	1	0	1	1	1	0	0	0	1	0	1	0	1	0	0	1	11	00	10	11
1	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1	0	00	11	11	00
2	0	0	0	1	0	1	0	0	1	1	0	1	0	0	1	0	00	00	01	10
3	0	1	1	0	1	0	0	0	1	0	1	0	0	1	0	0	00	11	10	00
4	0	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1	00	11	01	01
5	1	1	1	0	1	0	0	0	1	0	1	0	1	1	0	0	11	11	10	00
6	1	1	0	1	0	1	0	0	1	1	0	1	1	1	1	0	11	11	01	10
7	1	0	1	0	1	1	0	0	0	1	1	1	1	0	1	1	11	00	11	01
8	1	1	1	0	1	0	1	1	1	0	1	0	0	0	0	0	10	10	10	00
9	0	0	1	0	0	1	1	1	1	1	0	1	1	1	0	1	01	01	10	01
10	1	0	0	1	1	0	1	1	1	0	1	0	0	1	1	1	10	01	01	11
11	0	1	0	1	0	0	1	1	0	0	0	0	1	0	0	1	01	10	00	11
12	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	0	01	10	00	00
13	0	0	1	1	0	0	1	1	0	0	0	0	1	1	1	1	01	01	11	11
14	1	1	0	1	0	1	1	1	1	1	0	1	0	0	1	0	10	10	01	10
15	1	0	1	1	1	0	1	1	1	0	1	0	0	1	0	1	10	01	10	11

Next, at *Step 7*, we check the formation of check tests for the transformation elements of the values of the functions f_4 and f_3. The check tests are formed (see Table 3). At *Step 8*, we determine the values of the functions $g_i(X) = f_i(X) \oplus h_i(X), i \in \{3, 4, 5, 6\}$.

Then we optimize the functions g_3, …, g_6 (*Step 9*). Here are only the numbers of the allowed sets for the functions g_3, …, g_6: $g_6(X) = \{0, 5, 6, 7, 9, 11, 12, 13\}$, $g_5(X) = \{1, 3, 4, 5, 6, 9, 10, 13, 15\}$, $g_4(X) = \{1, 2, 4, 6, 7, 10, 13, 14\}$, $g_3(X) = \{0, 4, 7, 9, 10, 11, 13, 15\}$. (Alternatively, for each of these functions, it is possible to write all conjunctions on which they take unit values.) The subsequent optimization steps are trivial: each function can be optimized individually or as a system of Boolean functions [31]. At *Step 10*, the components are selected and the CED circuit is designed. The actions of this step need no detailed description. ♦

Now, we define the implementation complexity indicator L of the CED circuit in a preselected metric (e.g., the number of internal element inputs or the number of transistors used in the device on particular components). This indicator can be compared with that of the CED circuit designed by the duplication method, denoted by L_D. If $L < L_D$, then the method proposed in this paper is more efficient than duplication when ensuring the self-checking of both structures. Otherwise, another partitioning method is chosen, and the steps of the algorithm are repeated.

Due to formula (1), the time complexity of the proposed algorithm is asymptotically estimated by the value $2^{O(t)}$: the design problem is solved in exponential time with a linear exponent.

The presented algorithm is based on the following property of the *WS*(4, 2, 4)-code with the weights [2, 2, 2, 3]: for each check vector, four data vectors are formed once with the high bits 00, 01, 10, and 11, respectively. Note that *WS*(4, 2, 4)-codes with the weights [1, 1, 1, 2], [1, 1, 2, 3], and [1, 2, 2, 3] have the same property. These codes can also be used together with the proposed algorithm for designing the "base" structure (see Section 4). For other variants of weighting data symbols, a similar algorithm can be developed, e.g., by considering the non-repeatability of the other two data symbols in the data vectors for each check vector. For example, the *WS*(4, 2, 4)-codes with the weights [1, 1, 1, 2], [1, 1, 2, 2], [1, 1, 2, 3], [1, 2, 2, 2], [2, 2, 2, 3], [2, 2, 3, 3], and [2, 3, 3, 3] have a characteristic feature: for each check vector, four data vectors are formed, where each of the second and third high bits once take values 00, 01, 10, and 11, respectively.

Also, we mention an important fact: during its operation, the algorithm does not cover any error combinations at the outputs of the object under diagnosis. However, when designing a self-checking device, it is necessary to preselect implementation components, a fault model, and appropriate design methods [1, 11, 20, 21]. The problem of covering errors caused by

faults from a given model is solved by various methods, e.g., by transforming the structures into controllable ones with a given code or by selecting controllable groups of outputs [32, 33].

5. CED CIRCUIT DESIGN FOR DEVICES WITH MORE THAN SIX OUTPUTS

For multi-output devices with $n > 6$ outputs, the CED circuit is designed in accordance with the structure presented in Fig. 4. In this case, the set of outputs $W = \{f_1, f_2, ..., f_{n-1}, f_n\}$ of the source device is divided into subsets W_1, W_2, ..., W_{q-1}, W_q of cardinality 6, $q = \left\lceil \frac{n}{6} \right\rceil$. If $n(\text{mod } 6) = 0$, all q groups will contain six non-repeating outputs each; otherwise, $(q-1)$ groups will contain six non-repeating outputs each, and the last group is formed from the outputs $W_q = \{f_{n-5}, f_{n-4}, f_{n-3}, f_{n-2}, f_{n-1}, f_n\}$. For each group of six outputs of the object under diagnosis, the structure in Fig. 3 is designed by the above algorithm. The outputs of each TSC_1, ..., TSC_q are united at the inputs of the self-checking comparator qTRC1, which consists of $(q-1)$ TRC blocks.

The resulting self-checking device can be compared in terms of implementation complexity, e.g., with the device based on duplication. For this purpose, we define the implementation complexity indicator of the self-checking device (see above) in the given metric: $L = \sum_{i=1}^{q} L_i$. Then the efficiency is evaluated in comparison with the duplication method.

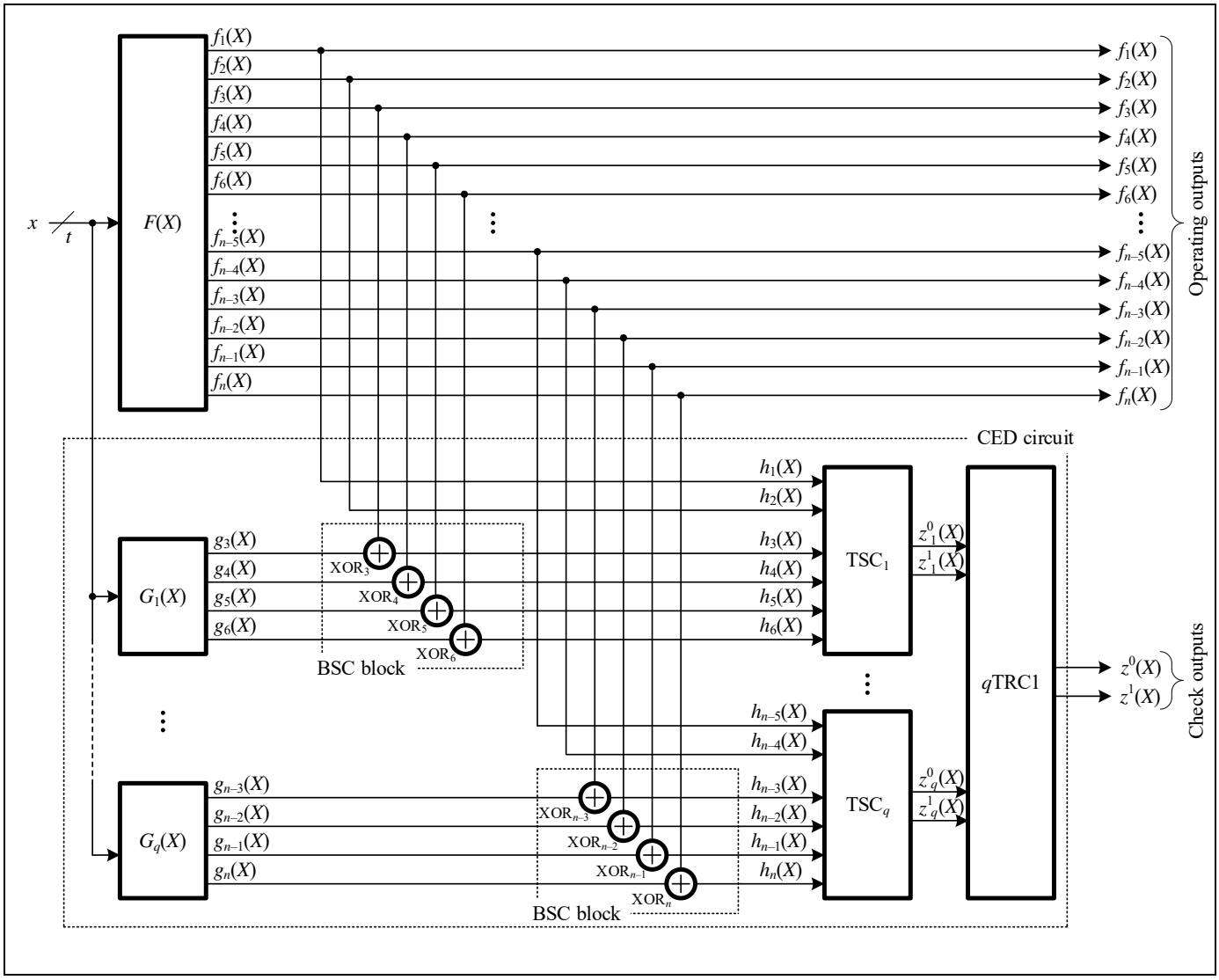

Fig. 4. The structural diagram of a CED circuit for devices with $n > 6$ based on BSC with transformation of the values of some functions implemented at the outputs of an object under diagnosis that form data symbols of the *WS*(4, 2, 4)-code.

Note that there are many ways to select groups of outputs into subsets of cardinality 6. The outputs for the set W_1 can be selected in C_n^6 ways. The outputs for the set W_2 can be selected in C_{n-6}^6 ways, and so on. Thus, for an arbitrary $j \in \{1, q-1\}$, we have $C_{n-6(j-1)}^6$ possible ways. (The set W_q is chosen uniquely.) The total number of ways to build the CED circuit is $\prod_{j=1}^{q-1} C_{n-6(j-1)}^6$. This value can be rewritten as

$$\begin{aligned}\prod_{j=1}^{q-1} C_{n-6(j-1)}^6 &= C_n^6 C_{n-6}^6 C_{n-12}^6 \cdot \ldots \cdot C_{n-6(q-3)}^6 C_{n-6(q-2)}^6 \\ &= \frac{n!}{6!(n-6)!} \cdot \frac{(n-6)!}{6!(n-12)!} \cdot \frac{(n-12)!}{6!(n-18)!} \cdot \ldots \\ &\cdot \frac{(n-6(q-3))!}{6!(n-6(q-2))!} \cdot \frac{(n-6(q-2))!}{6!(n-6(q-1))!} \\ &= \frac{n!}{(6!)^{q-1}(n-6q+6)!}.\end{aligned} \quad (2)$$

Since $q = \left\lceil \frac{n}{6} \right\rceil$, the expression (2) can be represented as a function of the number n only:

$$\prod_{j=1}^{\left\lceil \frac{n}{6} \right\rceil - 1} C_{n-6(j-1)}^6 = \frac{n!}{(6!)^{\left\lceil \frac{n}{6} \right\rceil - 1}\left(n - 6\left\lceil \frac{n}{6} \right\rceil + 6\right)!}. \quad (3)$$

For example, for a device with n = 20 outputs, formula (3) gives the following number of ways to select the subsets:

$$\begin{aligned}\prod_{j=1}^{\left\lceil \frac{20}{6} \right\rceil - 1 = 3} C_{20-6(j-1)}^6 &= \frac{20!}{(6!)^{\left\lceil \frac{20}{6} \right\rceil - 1}\left(20 - 6\left\lceil \frac{20}{6} \right\rceil + 6\right)!} \\ &= \frac{20!}{(6!)^3 2!} = 3\,259\,095\,840.\end{aligned}$$

These formulas indicate the total number of ways to select the subsets of outputs considering the minimum number of groups required for complete coverage without selecting the same check outputs in different groups (possibly, except for the last group). Including the same outputs in different groups for checking may be necessary to detect the required combinations of distortions in the outputs. Anyway, due to this circumstance, the number of ways to organize the CED circuit can be increased in comparison with the one yielded by the above algorithm and formula (3).

Concluding this section, we draw the reader's attention to the fact that, by the expression (3), all groups are analyzed in factorial time. (In other words, the time complexity is estimated by the value $O(n!)$.)

6. RESULTS OF EXPERIMENTS WITH TEST COMBINATIONAL CIRCUITS

Experiments with test combinational circuits from MCNC Benchmarks [34] were carried out to verify the efficiency of the proposed CED circuit design method based on BSC with *WS*(4, 2, 4)-codes. For each benchmark, the implementation complexity indicators were experimentally estimated for the devices with the CED circuit designed in accordance with the structure in Fig. 4. The SIS interpreter [35, 36] was used to optimize the logic correction functions, and the block $G(X)$ was designed. The implementation complexity indicators were determined in conventional units of the stdcell2_2.genlib library (the area occupied by the device on a chip). Some experimental results are summarized in Table 4 and additionally presented in Fig. 5. For each benchmark, the table provides the values of the corresponding parameters (t and n, the numbers of inputs and outputs), the number of selected groups (q), as well as the values of the implementation complexity indicators ($L_{F(X)}$, $L_{G(X)}$, and L, the conditional implementation complexity indicators of the blocks $F(X)$, $G(X)$ and the device with the CED circuit, respectively). For comparison, the column L_D presents the value of the implementation complexity indicator of the device designed by the duplication method; the last column of the table, the share δ of the implementation complexity indicator of the device with the CED circuit designed by the presented method in the corresponding indicator of the device with the CED circuit designed by the duplication method:

$$\delta = \frac{L}{L_D} \cdot 100\%.$$

No permutations of the outputs of the benchmarks were performed during the experiments. The groups of outputs were selected in their order in the description of the benchmark. For each group of outputs, the CED circuit was built, the blocks $G_j(X)$, $j = \overline{1, q}$, were jointly implemented, and the structure redundancy indicators were estimated.

The following results were obtained for the 20 benchmarks presented in Table 4 and Fig. 5. In 18 cases, the structure redundancy indicator decreased in comparison with duplication. The value of δ = 81.729% was calculated on the average for the 20 benchmarks considered. Therefore, the proposed CED

circuit design method has high efficiency in terms of structure redundancy.

For some benchmarks, arbitrary selection of groups of outputs without any output permutations within the groups gave no possibility to implement a self-checking CED circuit with forming all test combinations for the transformation elements and checker. However, simple permutation procedures for outputs within groups provide an effective solution. Here we describe the results for the dc1 circuit.

Initially, the outputs of the benchmarks were not permuted. For the dc1 circuit, the resulting distribution of test combinations for the CED circuit elements is presented in Table 5. Obviously, the number of test combinations has an uneven distribution, which is certainly due to the peculiarities of the dc1 circuit. (Besides the various arrangements of zero and one at the outputs, this circuit has zero at all outputs for the input combinations <1010> ... <1111>.) Also, the test combination <01> is not formed for the XOR_4 gate. With a simple permutation of the outputs f_5 and f_4 without changing the way to build the CED, the test combinations were redistributed, and the combination <01> was formed for the element XOR_4 in the CED circuit for the first group of outputs. The number of test combinations for TSC_1 and TSC_2 in both cases remained the same since only two outputs were permuted before the determination stage of the codewords of the *WS*(4, 2, 4)-code.

The output permutation also affected the structure redundancy of the self-checking device. In the first (original) case, the joint implementation complexity indicator of the blocks $G_j(X)$ was equal to 696 conditional units of the stdcell2_2.genlib library; in the second case, this indicator became equal to 672 conditional units of the stdcell2_2.genlib library. This decreased the value of L from 2872 to 2848 and the value of δ from 89.303% to 88.557% for this CED circuit design. This decrease is not significant.

Due to a large number of possible output permutations and ways to select controlled groups, one can form check tests for all elements of the CED circuit and regulate the structure redundancy indicators of the self-checking device.

Table 4

Experimental results

No.	Benchmark	t	n	q	$L_{F(X)}$, cond. units	$L_{G(X)}$, cond. units	L, cond. units	L_D, cond. units	δ, %
1	dc1	4	7	2	976	696	2872	3216	89.303
2	dekoder	4	7	2	736	752	2688	2736	98.246
3	wim	4	7	2	712	656	2568	2688	95.536
4	newbyte	5	8	2	592	680	2472	2656	93.072
5	p82	5	14	3	2368	1712	5976	7456	80.15
6	m1	6	12	2	3064	880	5144	8432	61.006
7	newapla2	6	7	2	600	592	2392	2464	97.078
8	sqr6	6	12	2	2648	2184	6032	7600	79.368
9	inc	7	9	2	2376	1792	5368	6432	83.458
10	newcpla2	7	10	2	1896	1440	4536	5680	79.859
11	max128	7	24	4	20 192	2520	25 304	45 184	56.002
12	m2	8	16	3	10 096	3024	15 016	23 328	64.369
13	m3	8	16	3	13 464	3744	19 104	30 064	63.544
14	m4	8	16	3	18 704	7152	27 752	40 544	68.449
15	mlp4	8	8	2	7224	9224	17648	15 920	110.854
16	tms	8	16	3	6784	3032	11 712	16 704	70.115
17	dk27	9	9	2	528	1168	2896	2736	105.848
18	max512	9	6	1	9632	5624	15 760	20 320	77.559
19	newcpla1	9	16	3	2520	2528	6944	8176	84.932
20	newxcpla1	9	23	4	3760	2832	9184	12 112	75.826
Average value									81.729

Table 5

The number of test combinations for CED circuit elements for the dc1 circuit

Group number	Gate	Variant I				Variant II			
		00	01	10	11	00	01	10	11
1	XOR_6	5	7	1	3	5	7	1	3
	XOR_5	3	6	2	5	3	6	2	5
	XOR_4	9	0	3	4	8	1	2	5
	XOR_3	6	4	5	1	6	4	5	1
	TSC_1	6	2	1	7	6	2	1	7
2	XOR_7	4	6	2	4	4	6	2	4
	XOR_6	5	7	1	3	5	7	1	3
	XOR_5	6	3	3	4	7	2	4	3
	XOR_4	6	3	5	2	7	2	6	1
	TSC_2	6	4	2	4	6	4	2	4

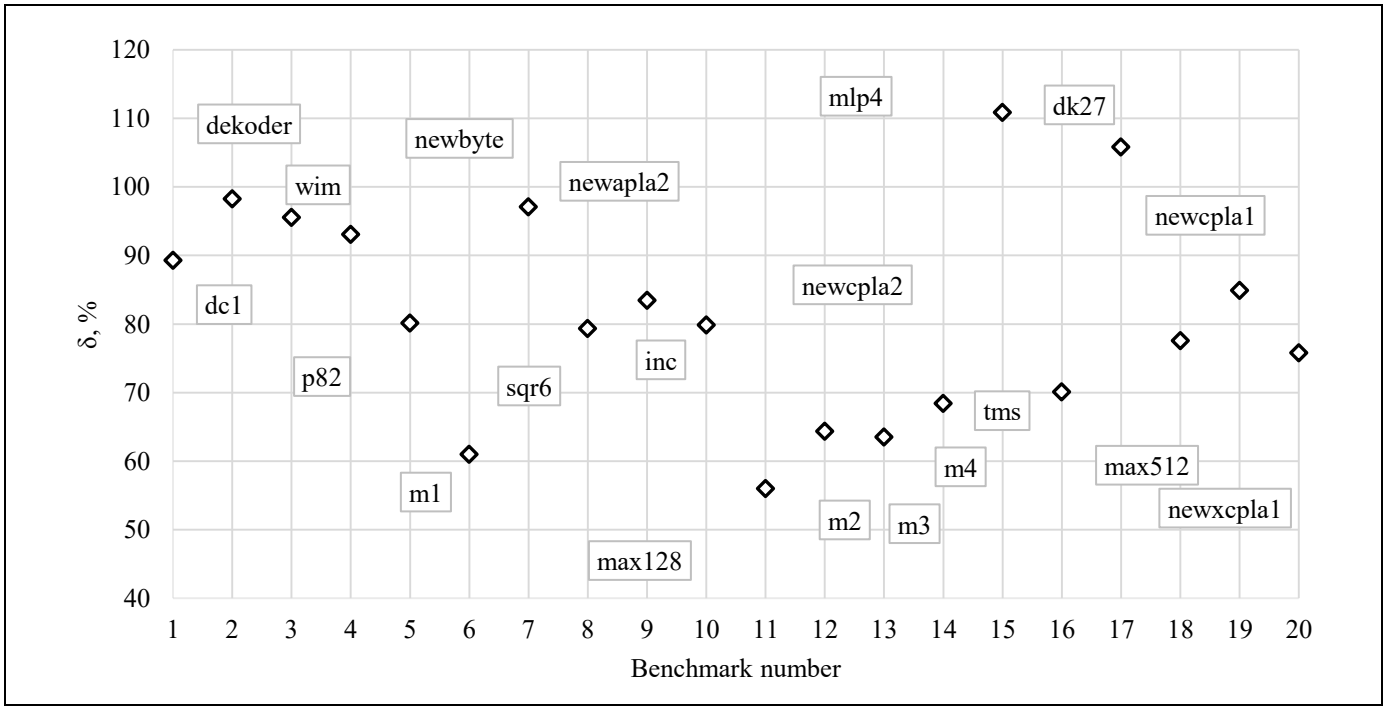

Fig. 5. The graphical representation of the experimental results.

Further studies and experiments may aim to minimize the value of the indicator *L*, to ensure the checkability of all CED circuit elements, and to obtain a uniform distribution of test combinations for encoders and transformation elements in CED circuit structures. These tasks go beyond the scope of the paper.

CONCLUSIONS

The CED circuit design method with BSC using *WS*(4, 2, 4)-codes and the formation of the "base" structure of a self-checking device, the method proposed in this paper, provides the designer of such devices with higher variability compared to the method known previously, in which correction applies in the CED circuit only to the functions participating in the formation of check code symbols. In turn, the designer can optimize the indicators of structure redundancy, controllability, energy consumption, etc. for the self-checking devices being built.

According to the experimental results presented above, in many cases it is possible to reduce the structure redundancy indicators in comparison with the standard duplication method; for about half of the cir-

cuits, it is possible to achieve δ = 50–70%. Therefore, the method can be applied in practice to implement self-checking digital devices.

The method has the following advantages: if there is a possibility to build a self-checking device by this method, it will yield results. This circumstance follows from the need to check, for each function implemented at the outputs of the device $F(X)$, whether it takes value 0 (and 1) for at least two sets of argument values. If this condition fails, a self-checking device will not be designed even using standard methods (e.g., duplication): a check test for the comparator will not be formed. In addition, the method surely forms test combinations for the first two transformation elements out of four. Among the drawbacks of the method, we note that all combinations included in the check test are not necessarily formed for the remaining two transformation elements; see the experimental results for several test circuits. In this case, it is necessary to permute the outputs and, possibly, to regroup them.

Also, we emphasize that the proposed "base" structure yields a large number of variants to form the values of correction functions whereas the presented algorithm only one of a few. This fact makes application of the method promising, particularly with other algorithms for designing self-checking devices.

An interesting development of the method presented above is a generalized CED circuit design method with BSC using weight-based Bose–Lin codes with an arbitrary value of m and different values of the moduli $M \in \left\{2^2, 2^3, ..., 2^{\lceil \log_2(m+1) \rceil}\right\}$. There are at least two lines of related research. The first one is to study the peculiarities of applying a sequence of natural number series weights when constructing the code: the presence of weights representing multiples of the modulus value leads to the appearance and, as m grows, to an increase in the number of single undetectable errors [21]. The second line is to study the peculiarities of applying sequences of arbitrary natural number weights.

Note finally that the use of BSC in the design of self-checking CED circuits is an unexplored approach to building devices and systems with fault detection, which may significantly improve their efficiency compared to known methods.

REFERENCES

1. Sogomonyan, E.S. and Slabakov, E.V., *Samoproveryaemye ustroistva i otkazoustoichivye sistemy* (Self-Checking Devices and Fault-Tolerant Systems), Moscow: Radio i Svyaz', 1989. (In Russian.)
2. Lala, P.K., *Self-Checking and Fault-Tolerant Digital Design*, San Francisco: Morgan Kaufmann Publishers, 2001.
3. Gharibi, W., Hahanov, V., Chumachenko, S., et al., Vector-Logic Computing for Faults-As-Address Deductive Simulation, *IAES International Journal of Robotics and Automation*, 2023, vol. 12, no. 3, pp. 274–288. DOI: 10.11591/ijra.v12i3.pp274-288.
4. Ubar, R., Raik, J., Jenihhin, M., and Jutman, A., *Structural Decision Diagrams in Digital Test: Theory and Applications*, Cham: Springer Nature, 2024. DOI: 10.1007/978-3-031-44734-1.
5. Parkhomenko, P.P. and Sogomonyan, E.S., *Osnovy tekhnicheskoi diagnostiki. Tom 2: Optimizatsiya algoritmov diagnostirovaniya, apparaturnye sredstva)* (Foundations of Technical Diagnosis. Vol. 2: Optimization of Diagnostic Algorithms, Hardware Means), Moscow: Energoatomizdat, 1981. (In Russian.)
6. Drozd, A.V., Kharchenko, V.S., and Antoshchuk, S.G., *Rabochee diagnostirovanie bezopasnykh informatsionno-upravlyayushchikh sistem* (Operational Diagnosis of Safe Information and Control Systems), Khar'kov: Zhukovskii National Aerospace University, 2012. (In Russian.)
7. Mikoni, S., Top Level Diagnostic Models of Complex Objects, in *Lecture Notes in Networks and Systems*, 2022, vol. 442, pp. 238–249. DOI: 10.1007/978-3-030-98832-6_21.
8. Bennetts, R.G., *Design of Testable Logic Circuits*, Boston: Addison-Wesley, 1984.
9. McCluskey, E.J., *Logic Design Principles (with Emphasis on Testable Semicustom Circuits)*, New Jersey: Prentice-Hall, 1986.
10. Abramovici, M., Breuer, M.A., and Friedman, A.D., *Digital System Testing and Testable Design*, New Jersey: IEEE Press, 1998.
11. Göessel, M., Ocheretny, V., Sogomonyan, E., and Marienfeld, D., *New Methods of Concurrent Checking*, 1st ed., Dordrecht: Springer Science+Business Media, 2008.
12. Sahana, A.R., Chiraag, V., Suresh, G., et al., Application of Error Detection and Correction Techniques to Self-Checking VLSI Systems: An Overview, *Proceedings of 2023 IEEE Guwahati Subsection Conference (GCON)*, Guwahati, 2023. DOI: 10.1109/GCON58516.2023.10183449.
13. Mitra, S. and McCluskey, E.J., Which Concurrent Error Detection Scheme to Choose?, *Proceedings of International Test Conference*, Atlantic City, 2000, pp. 985–994. DOI: 10.1109/TEST.2000.894311.
14. Sagalovich, Yu.L., and Solomennikov, V.Yu., Fault Detection in Network Realizations of Systems of Monotone Boolean Functions, *Problems Inform. Transmission*, 1997, vol. 33, no. 2, pp. 163–173.
15. Gessel', M., Dmitriev, A.V., Sapozhnikov, V.V., and Sapozhnikov, Vl.V., A Functional Fault-Detection Self-test for Combinational Circuits, *Automation and Remote Control*, 1999, vol. 60, no. 11, pp. 1653–1663.
16. Efanov, D.V. and Pogodina, T.S., Properties Investigation of Self-Dual Combinational Devices with Calculation Control Based on Hamming Codes, *Informatics and Automation*, 2023, vol. 22, no. 2, pp. 349–392. DOI: 10.15622/ia.22.2.5. (In Russian.)
17. Efanov, D.V., The Self-Checking Structures Implementation Features Based on the Inverting Data and Linear Block Code Method, *Tomsk State University Journal of Control and Computer Science*, 2023, no. 65, pp. 126–138. DOI: 10.17223/19988605/65/13. (In Russian.)
18. Goessel', M., Morozov, A.V., Sapozhnikov, V.V., and Sapozhnikov, Vl.V., Checking Combinational Circuits by the

Method of Logic Complement, *Automation and Remote Control*, 2005, vol. 66, no. 8, pp. 1336–1346.
19. Goessel, M. and Graf, S., *Error Detection Circuits*, London: McGraw-Hill, 1994.
20. Sapozhnikov, V.V., Sapozhnikov, Vl.V., and Efanov, D.V., *Kody s summirovaniem dlya sistem tekhnicheskogo diagnostirovaniya. Tom 1: Klassicheskie kody Bergera i ikh modifikatsii* (Sum Codes for Technical Diagnosis Systems. Vol. 1: Classical Berger Codes and Their Modifications), Moscow: Nauka, 2020. (In Russian.)
21. Sapozhnikov, V.V., Sapozhnikov, Vl.V., and Efanov, D.V., *Kody s summirovaniem dlya sistem tekhnicheskogo diagnostirovaniya. Tom 2: Vzveshennye kody s summirovaniem*, (Sum Codes for Technical Diagnosis Systems. Vol. 2: Weight-Based Codes), Moscow: Nauka, 2021. (In Russian.)
22. Aksenova, G.P., Necessary and Sufficient Conditions for Design of Completely Checkable Modulo 2 Convolution Circuits, *Automation and Remote Control*, 1979, vol. 40, no. 9, pp. 1362–1369.
23. Piestrak, S.J., *Design of Self-Testing Checkers for Unidirectional Error Detecting Codes*, Wrocław: Oficyna Wydawnicza Politechniki Wrocławskiej, 1995.
24. Das, D. and Touba, N.A., Synthesis of Circuits with Low-Cost Concurrent Error Detection Based on Bose–Lin Codes, *Journal of Electronic Testing: Theory and Applications*, 1999, vol. 15, no. 1-2, pp. 145–155. DOI: 10.1023/A:1008344603814.
25. Efanov, D.V., Sapozhnikov, V.V., and Sapozhnikov, Vl.V., The Self-Checking Concurrent Error-Detection Systems Synthesis Based on the Boolean Complement to the Bose–Lin Codes with the Modulo Value M = 4, *Electronic Modeling*, 2021, vol. 43, no. 1, pp. 28–45. DOI: 10.15407/emodel.43.01.028.
26. Efanov, D.V., Sapozhnikov, V.V., and Sapozhnikov, V.V., Sum Code Family Formation Method with Undetectable Error Minimum in Data Vectors, *Informatics*, 2019, vol. 16, no. 3, pp. 101–118. (In Russian.)
27. Sapozhnikov, V.V. and Sapozhnikov, Vl.V., *Samoproveryaemye diskretnye ustroistva* (Self-Checking Discrete Devices), St. Petersburg: Energoatomizdat, 1992. (In Russian.)
28. Efanov, D.V., Sapozhnikov, V.V., Sapozhnikov, and Vl.V., On Summation Code Properties in Functional Control Circuits, *Automation and Remote Control*, 2010, vol. 71, no. 6, pp. 1117–1123.
29. Pashukov, A.V., Application of Weight-Based Sum Codes at the Synthesis of Circuits for Built-in Control by Boolean Complement Method, *Transport Automation Research*, 2022, no. 1. pp. 101–114. DOI: https://doi.org/10.20295/2412-9186-2022-8-1-101-114. (In Russian.)
30. Carter, W.C., Duke, K.A., and Schneider, P.R., US Patent 3559167A, 1971.
31. Zakrevskii, A.D., Pottosin, Yu.V., and Cheremisinova, L.D., *Logicheskie osnovy proektirovaniya diskretnykh ustroistv* (Logical Foundations of Designing Discrete Devices), Moscow: Fizmatlit, 2007. (In Russian.)
32. Morosow, A., Sapozhnikov, V.V., Sapozhnikov, Vl.V., and Goessel, M., Self-Checking Combinational Circuits with Unidirectionally Independent Outputs, *VLSI Design*, 1998, vol. 5, no. 4, pp. 333–345. DOI: 10.1155/1998/20389.
33. Efanov, D.V., Sapozhnikov, V.V., and Sapozhnikov, Vl.V., Organization of a Fully Self-Checking Structure of a Combinational Device Based on Searching for Groups of Symmetrically Independent Outputs, *Automatic Control and Computer Sciences*, 2020, vol. 54, no. 4, pp. 279–290. DOI: 10.3103/S0146411620040045.
34. *Collection of Digital Design Benchmarks*. URL: https://ddd.fit.cvut.cz/www/prj/Benchmarks/. (Accessed February 24, 2024.)
35. Sentovich, E.M., Singh, K.J., and Moon, C., Sequential Circuit Design Using Synthesis and Optimization, *Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers & Processors*, Cambridge, 1992, pp. 328–333. DOI: 10.1109/ICCD.1992.276282.
36. Sentovich, E.M., Singh, K.J., and Lavagno, L., *SIS: A System for Sequential Circuit Synthesis*, Berkeley: Electronics Research Laboratory, Department of Electrical Engineering and Computer Science, University of California, 1992.

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

Received March 15, 2024, and revised August 24, 2024. Accepted August 29, 2024.

Author information

Efanov, Dmitry Viktorovich. Dr. Sci. (Eng.), Peter the Great Saint Petersburg Polytechnic University, St. Petersburg, Russia; Russian University of Transport, Moscow, Russia
✉ TrES-4b@yandex.ru
ORCID iD: https://orcid.org/0000-0002-4563-6411

Yelina, Yeseniya Igorevna. Postgraduate, Peter the Great Saint Petersburg Polytechnic University, St. Petersburg, Russia
✉ eseniya-elina@mail.ru
ORCID iD: https://orcid.org/0009-0004-4167-3591

Cite this paper

Efanov, D.V. and Yelina, Y.I., Design of Self-Checking Digital Devices with Boolean Signals Correction Using Weight-Based Bose–Lin Codes. *Control Sciences* **4**, 22–36 (2024). http://doi.org/10.25728/cs.2024.4.3

Original Russian Text © Efanov, D.V., Yelina, Y.I., 2024, published in *Problemy Upravleniya*, 2024, no. 4, pp. 26–43.

Translated into English by *Alexander Yu. Mazurov*, Cand. Sci. (Phys.–Math.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com

DOI: http://doi.org/10.25728/cs.2024.4.4

AN ENTROPY-BASED COMPOSITE INDICATOR FOR EVALUATING THE EFFECTIVENESS OF RECOMMENDER SYSTEM ALGORITHMS

R. S. Kulshin* and A. A. Sidorov**

Tomsk State University of Control Systems and Radioelectronics, Tomsk, Russia

*✉ roman.s.kulshin@tusur.ru, **✉ anatolii.a.sidorov@tusur.ru

Abstract. The problem of forming a composite indicator for evaluating the effectiveness of recommender system algorithms is considered. A novel composite indicator is proposed by combining individual metrics using the entropy method. The testing base of this study consists of 12 algorithms (on the one hand) and 3 datasets (on the other). For each algorithm–dataset combination, we calculate partial criteria used in evaluating recommender systems. According to the results presented below, the composite indicator is an effective tool for evaluating the performance of recommender system algorithms. As is shown, the performance of the algorithms varies depending on the size and other basic characteristics of a particular dataset. This indicator can be used to develop more efficient algorithms and their ensembles as well as to optimize hyperparameters and improve the quality of recommendations.

Keywords: recommender systems, composite indicator, algorithms, metrics, datasets.

INTRODUCTION

In a hybrid world, when real social relations are closely intertwined with digital practices of communication and economic activities, recommender systems are becoming in-demand services between various business actors, both the suppliers and consumers of different types of goods and services in a market. For instance, under the constant growth of data volumes, the recipients of various benefits face the need for effective search and filtering of information [1], and their producers are looking for ways to improve the effectiveness of marketing activities to stimulate the consumption of goods and services. In this context, recommender systems provide personalized recommendations in various business segments: from online shopping to music and movie streaming.

In e-commerce, recommendations allow shoppers to find goods matching their interests and preferences. Recommender systems analyze data on previous purchases, goods views [2], and behavioral patterns to suggest the most relevant options. On music and movie streaming platforms, recommendations play a key role when meeting users' interests: personalized playlists and content recommendations are provided based on the analysis of preferences for genres, artists, and film directors and the consideration of ratings and reviews [3].

The technologies underlying recommender systems include machine learning as well as collaborative and content filtering algorithms. Many of them also involve the neural network approach [4] to analyze and predict user preferences more accurately. The demand for such software services and products results in the emergence of many algorithms with two parameters as their effectiveness evaluation metrics: time and memory, characterized by an asymptotic constraint. Big O is the most common method for evaluating algorithms [5]. The Big O notation describes the complexity of algorithms and allows comparing their effectiveness based on homogeneous data.

At the same time, many other metrics characterize the performance of recommendation algorithms. The list of evaluation criteria is quite extensive and includes both typical parameters of machine learning methods and specialized metrics of recommendation algorithms [6]. They measure the accuracy, completeness, diversity, reliability, and understandability of

recommendations, user satisfaction, catalog coverage, query processing speed, and other characteristics. Each metric has its advantages and limitations depending on the context of application and the goals of the recommender system.

The use of various metrics contributes to a better understanding of, first, the generalized aspects of recommendation algorithms that often have no common meaning for the scientific and professional community (performance, efficiency, effectiveness, quality) and, second, the suitability of algorithms for particular tasks and audiences.

In this situation, the following difficulties arise when comparing algorithms and selecting an appropriate one to solve a particular task:

• There are many metrics for evaluating the algorithms underlying recommender systems, each being suitable for different aspects of recommendations. This may produce ambiguous results when different metrics point to different algorithms as the best ones.

• Some metrics may be mutually related or dependent. An improvement in the value of one metric may worsen the value of another one. This complicates the choice of an appropriate algorithm.

• The values of metrics may vary depending on the context. For example, accuracy may differ in different fields of application.

Creating a unified evaluation metric for a recommendation algorithm is a necessary step to ensure the objectivity and comparability of recommendations as well as to improve their quality. This step serves to standardize the evaluation of algorithms' efficiency, select the most appropriate one for a particular task, and optimize resource utilization.

The main hypothesis of this paper is that all recommendation algorithms behave differently depending on the data context, the size of datasets, and other characteristics of the recommendation environment. To avoid the time-consuming empirical selection of an algorithm with a comparison of many metrics for a particular situation, we propose to create a supercriterion for evaluating recommendation algorithms.

1. METHODOLOGY

1.1. Composite Indicator as a Generalized Measure

As a single criterion, we introduce a composite indicator representing an aggregate combining several relevant metrics to measure a complex synthetic construct [7–10]. Nowadays, composite indicators are applied in various fields for evaluating:

– the productivity of companies, markets, and the economy as a whole [11–13],

– the patient's health status [14–16],

– environmental conditions [17–19],

– risks in various fields [20–22],

– the quality of education and student performance [23–25],

– et al.

They have advantages and drawbacks. For instance, the advantages of composite indicators include the following:

• Several individual indicators or variables are combined into a generalized measure, which is convenient for analysis and comparison [7, 9, 10, 24].

• Analysis is conducted based on the logical meaningful union of variables related to a particular field or topic [9, 10, 13].

• Composite indicators can be used in many fields and for different tasks, which makes them versatile tools for analysis and prediction [9, 13].

Speaking about the merits of composite indicators, we have to mention some of their disadvantages:

• The bottleneck of this methodology is the assignment of weights to individual indicators: there exists the possibility of manipulation when justifying the decision based on the calculation results of a given composite indicator [7, 24].

• The use of composite indicators may be limited by the availability of data required to justify the supercriterion structure in substantive terms [7, 13].

Despite these drawbacks, composite indicators can be effective because their applicability in many contexts makes them flexible tools adaptable to different tasks and conditions.

Application of the methodological approach based on calculating a composite indicator is quite disputable, which has led to the emergence of two camps in the scientific community: its opponents and supporters. Despite the diversity of views and controversial points (e.g., selecting the initial list of criteria, methods for determining weights and aggregation, and ways of normalizing indicators measured in different ranges and scales), the composite indicator is a rather convenient and well-interpretable tool for comparing alternatives and making decisions under multiple criteria. Due to its high degree of "customizability," this method can be adapted to different situations and needs.

With the diverse fields of application of this methodological approach and the possibility of working with heterogeneous data, we can use it in a new field, i.e., for comprehensively evaluating the effectiveness of algorithms. By using the composite indicator, we can consider various performance aspects of algorithms and summarize them into a single metric, which seems more convenient when analyzing performance

results described by a set of partial criteria. This will significantly improve the objectivity of the final evaluation.

Composite indicators, successfully used for analysis and evaluation in socio-economic fields [11–25], have not been previously applied to evaluate algorithms. The approach proposed below is novel due to adapting the methodology to the field of information technology. This paper mainly focuses on evaluating the effectiveness of recommendation algorithms, which opens new opportunities for analyzing them and improving their performance.

1.2. Metrics

To form a composite indicator, we select the main evaluation metrics for machine learning algorithms and personalized recommender systems.

• *Average Popularity* (*AP*) is the average popularity of recommended elements (items, objects) [26, 27]:

$$AP = \frac{1}{|U|}\sum_{u\in U}\frac{\sum_{i\in R(u)}\phi(i)}{|R(u)|},$$

where U denotes the set of all users with generated recommendations; $|U|$ is the number of such users; u is a particular user whose data are employed in calculations; $R(u)$ is the set of recommendations; $|R(u)|$ is the number of elements in this set; $\phi(i)$ is the number of recommendations of element i in training data.

High values of *AP* indicate that the system frequently recommends popular elements, which can attract users' attention. On the other hand, excessively high values of this metric may deteriorate the diversity of recommendations.

• *Grouped Area under the Curve* (*GAUC*) characterizes the quality of ranking for all users:

$$GAUC = \frac{1}{\sum_{u\in U}|R(u)|}\sum_{u\in U}|R(u)|\,AUC(u).$$

Here $AUC(u)$ is a quality evaluation metric for models in classification and ranking problems:

$$AUC(u) = \left(|R(u)|(n+1) - \frac{|R(u)|(|R(u)|+1)}{2} - \sum_{i=1}^{|R(u)|} rank_i\right) \Big/ \left(|R(u)|(n-|R(u)|)\right),$$

where $rank_i$ denotes the descending rank of element i of the set $R(u)$; n is the total number of user interactions with the element.

AUC reflects the area under the *Receiver Operating Characteristic* (*ROC*) curve, a graph showing the ratio of the share of correctly classified positive cases to the share of incorrectly classified positive cases when varying the decision threshold [28]. This metric quantitatively estimates the model's capability to discriminate between classes (e.g., positive and negative cases) under different thresholds [29].

GAUC demonstrates how well the model ranks recommendations for users (in other words, how effectively the model distinguishes between recommendations satisfying the user and those not). This metric is applied for tasks where the ranking of suggestions is important, such as goods or content recommendations.

• *Gini Index* (*GI*) characterizes the diversity of the distribution of generated recommendations:

$$GI = \frac{\sum_{i=1}^{|I|}(2i-|I|-1)P(i)}{|I|\sum_{i=1}^{|I|}P(i)},$$

where $P(i)$ denotes the popularity of each element after training the algorithm; $|I|$ is the rating of recommended elements in non-decreasing order.

GI serves to measure how uniformly the recommendations are distributed for each user. A more uniform distribution means that the recommendations are more diverse and meet the different interests of the user. If the distribution is close to ideal, *GI* will take a value near 0. If the recommendations are centered around a small number of elements, *GI* will be closer to the maximum value [30].

• *HitRate* (*HR*) is the ratio of the number of recommendations satisfying the user to the total number of recommendations [31]:

$$HR = \frac{1}{|U|}\sum_{u\in U}|\overline{R(u)}\cap R(u)|,$$

where $\overline{R(u)}$ denotes the set of recommendations satisfying the user.

HR indicates what share of recommendations helpful to the user; the higher its value is, the more successful the recommender system will be.

• *Precision* (positive predictive value) is the share of relevant elements among all recommended ones [32]:

$$Precision = \frac{1}{|U|}\sum_{u\in U}\frac{|\overline{R(u)}\cap R(u)|}{|\overline{R(u)}|}.$$

This indicator shows how accurately the system selects the entities that will be preferable to the user. The higher value *Precision* takes, the more accurate the recommendations will be.

• *Recall* is a measure to calculate the share of relevant elements from the entire set of generated recommendations [33]:

$$Recall = \frac{1}{|U|}\sum_{u\in U}\frac{|\overline{R(u)}\cap R(u)|}{|R(u)|}.$$

It indicates what share of relevant elements the system can consider in the recommendations.

At first glance, *Recall* and *Precision* are similar metrics; meanwhile, they have fundamental differences. *Recall* focuses on detecting as many relevant elements as possible and minimizing missed opportunities; *Precision*, on the accuracy of element selection and minimizing irrelevant recommendations.

A high *Recall* value means that the system covers the user's interests well but may include more noise in the recommendations. A high *Precision* value means that the system provides accurate recommendations but may miss some interesting elements.

• *Shannon Entropy* (*SE*) is the variety of recommendations generated for a user [34]:

$$SE = -\sum_{i=1}^{|I|} p(i)\log p(i),$$

where $p(i)$ denotes the probability of recommending element i.

If recommendations are diverse and cover different user interests, entropy will be high. Entropy estimation in recommender systems can help optimize the balance between personalization (providing recommendations meeting user's unique interests) and diversity (providing recommendations covering a wider range of interests).

• *Mean Average Precision* (*MAP*) is the overall quality of elements ranking:

$$MAP@K = \frac{1}{|U|}\sum_{u\in U}\left(\frac{1}{\min(|\overline{R(u)}|,K)} \times \sum_{j=1}^{|\overline{R(u)}|} |I|\overline{R(u)}_j\ Precision\right),$$

$$\overline{R(u)}_j \in R(u),$$

where K is the truncated number of generated recommendations; j is the index of the recommendation satisfying the user.

The truncated number of generated recommendations is a sample of the most recommended elements, specified by the system developer. It can take any value not exceeding the number of recommended elements. A value of 10 is most widespread in practice.

This indicator is a helpful metric for evaluating the quality of recommender system models, especially when it is important to consider the ranking of recommendations. Each user is treated separately to consider individual preferences and interests. Since *MAP* averages the *Precision* value of all users, this metric provides a generalized evaluation of the recommender system ranking.

• *Mean reciprocal rank* (*MRR*) is the ranking quality of the first element in the list of recommended ones [35]:

$$MRR = \frac{1}{|U|}\sum_{u\in U}\frac{1}{rank_u^*},$$

where $rank_u^*$ is the rank position of the first relevant element found by the algorithm for the user.

This indicator is widespread to evaluate the quality of search engines, recommender systems, and other tasks where it is important to consider the ranking of results. The higher the *MRR* value is, the better the result will be. The *MRR* value will equal 1 if the relevant element is always at the first position in the ranked list.

• *Normalized discounted cumulative gain* (*NDCG*) is a ranking quality indicator that considers the position–relevance relationship of elements in a ranked list [36]:

$$NDCG@K = \frac{1}{|U|}\sum_{u\in U}\left(\frac{1}{\sum_{i=1}^{\min(|R(u)|,K)}\frac{1}{\log_2(i+1)}} \times \sum_{i=1}^{K}\delta(i\in R(u))\frac{1}{\log_2(i+1)}\right),$$

where δ denotes the indicator function (if $i \in R(u)$, then δ = 1; otherwise δ = 0); i is the recommendation included in the truncated list.

NDCG is widespread to evaluate the quality of ranking in recommender systems, especially when the presence of relevant recommendations and their order in the list are both important.

In addition, metrics indicating memory and time consumption are taken for experiments. The consumed RAM is calculated using the *memory_profiler* package of the Python language. The training time of an algorithm is that required to tune hyperparameters or train embedding layers based on the *Recall* metric. Hyperparameters can be tuned based on a single metric. The *Recall* metric is chosen because of its importance for problems where missing data may have serious consequences.

The calculations presented below were performed on a PC with a 2.40GHz Intel(R) Xeon(R) Silver 4214R CPU and 132 Gb RAM. When reproducing the results, the time measured during an experiment may vary depending on the computing capabilities.

2. THE MATERIALS AND TOOLS OF THIS STUDY

2.1. Datasets

The following conditions are taken into consideration when forming the datasets. To make the source material diverse and the study adequate, we use three datasets representing different amounts of information about user preferences. The first two datasets, MovieLens 100k and MovieLens 1m, include 100 thousand and 1 million records, respectively. Both datasets contain valuable information about the user ratings of movies and demographic data. MovieLens is used for research in recommender systems and machine learning. Amazon Gift Card, part of Amazon Review Data, is chosen as the third dataset. Belonging to a different field than MovieLens, it has high data sparsity. Amazon Review Data (2018) is a set of user responses from the Amazon online store, including related information (product, user, ratings, and response text). This set contains about 35 million user responses over 18 years and is most widespread in machine learning. Amazon Review Data (2018) can be applied for various purposes, such as response tone analysis, natural language processing, and model training. The Amazon Gift Card dataset, which includes information about gift card responses, is taken for research.

Table 1 summarizes the characteristics of these datasets.

Table 1

The characteristics of datasets

Parameter	Datasets		
	MovieLens 100k	MovieLens 1m	Amazon Gift Card
Number of users	944	6041	128 878
Average number of user actions	106.04	165.59	1.1421
Number of elements	1683	3707	1549
Average number of actions with an element	59.45	26988	95.08
Number of intersections	100 000	1 000 209	147 194
Data sparsity, %	93.70	95.50	99.92

Thus, the selected sets satisfy the conditions described above. At the same time, MovieLens 1m is a dataset with a larger number of records compared to MovieLens 100k, which is important to analyze the impact of the number of dataset records on the algorithm evaluation.

When comparing Amazon Gift Card to MovieLens, one observes a significant difference in the average number of user actions: a value slightly exceeding 1 in the former case and over 100 in the latter. This difference creates significant challenges for recommendation algorithms. When dealing with Amazon Gift Card and its relatively small number of user actions, recommendation algorithms face difficulties in predicting and generalizing user preferences. The truncated amount of data may lead to underrepresentative samples, making forecasts and recommendations less accurate.

2.2. Algorithms

The following recommendation algorithms are evaluated within this study:

• Bayesian Personalized Ranking from Implicit Feedback (BPR) [37];

• Large-Scale Information Network Embedding (LINE) [38];

• Neural Collaborative Filtering (NeuCF) [39];

• Deep Matrix Factorization (DMF) [40];

• Spectral Collaborative Filtering (SpectralCF) [41];

• Simplifying and Powering Graph Convolution Network for Recommendation (LightGCN) [42];

• Variational Autoencoders for Collaborative Filtering (MultiVAE) [43];

• Collaborative Denoising Auto-Encoders (CDAE) [44];

• Ranking-Critical Training for Collaborative Filtering (RaCT) [45];

• Sparse Linear Method (SLIM) [46];

• Item-based collaborative filtering (ItemKNN) [47];

• Diffusion Recommender Model (DiffRec) [48].

This list covers different recommendation algorithms for maximum diversity. Note that currently, there are many more approaches than are considered in this paper. The ones above are selected to adequately represent various algorithms and their features in the context of recommendation generation. At the same time, other algorithms also deserve attention and may be the subject of further research in this area.

RecBole, an open-source library developed in the Python programming language and the PyTorch machine learning framework, is chosen to implement the algorithms. This platform offers a wide range of algorithms and approaches to building recommendations, as well as tools for developing, testing, and evaluating recommendation algorithms [49].

3. THE RESULTS OF THIS STUDY

3.1. Calculation of Algorithm Performance

Calculations were performed for all the three datasets. For clarity, we describe intermediate stages only for MovieLens 100k. When collecting metrics, the parameter K (the truncated number of generated recommendations) was set equal to 10.

All collected metrics, except for memory, preparation time, prediction time, and *Average Popularity*, are coefficients taking values between 0 and 1. Memory consumed is represented in megabytes; time metrics, in seconds; *Average Popularity*, in the number of user interactions with an element. The collected data are shown in Tables 2–4.

3.2. Model

We form the composite indicator model as follows: the 13 parameters are convolved into four subindicators of the second layer; *Preparation time*, *Prediction time* (the time to generate recommendations), and *Memory* (the memory size consumed) are aggregated into the Resources subindicator; *Recall* and *Precision* are reduced to Accuracy; the *GAUC*, *MMR*, *NDCG*, *HitRate*, and *MAP* metrics are convolved into the Ranking subindicator; *Average Popularity*, *Gini Index*, and *Shannon Entropy* are generalized into the Diversity subindicator. Figure 1 presents the structure of the composite indicator.

This model is based on the principle of logical union of parameters.

3.3. Calculations

To create the composite indicator, three main tasks have to be solved:

• normalize the partial criteria, as they have different dimensions and units of measurement;

• calculate the weights on the network layers;

• determine the principle of convolving the partial criteria into the composite indicator and its structural elements.

For the convenience of building the composite indicator, the values of the partial criteria should meet the following requirements:

– All partial criteria should be dimensionless.

– To compare different elements with each other, the values of the partial criteria should vary within the same range, e.g., from 0 to 1.

– All partial criteria should be unidirectional.

We apply the minimax value normalization of the source to satisfy these requirements:

$$e_{ij} = \frac{x_{ij} - \min\limits_{k=1,n}(x_{kj})}{\max\limits_{k=1,n}(x_{kj}) - \min\limits_{k=1,n}(x_{kj})},$$

where e_{ij} is the normalized value of the jth metric for the ith algorithm; x_{ij} is the factual value of the jth metric for the ith algorithm; n is the number of algorithms.

For some indicators, such as *Preparation time*, *Prediction time*, *Memory*, and *Average Popularity*, the normalization procedure is applied with value inversion in the same numerical range [0, 1], since increasing their values worsens the resulting evaluation:

$$e_{ij} = 1 - \frac{x_{ij} - \min\limits_{k=1,n}(x_{kj})}{\max\limits_{k=1,n}(x_{kj}) - \min\limits_{k=1,n}(x_{kj})}.$$

Table 5 presents the normalized values of the indicators.

To form the composite indicator, we involve an entropy method for finding the weights of partial criteria. This method is based on analyzing the estimates of the standard deviations of partial criteria over the entire set of elements under consideration [50].

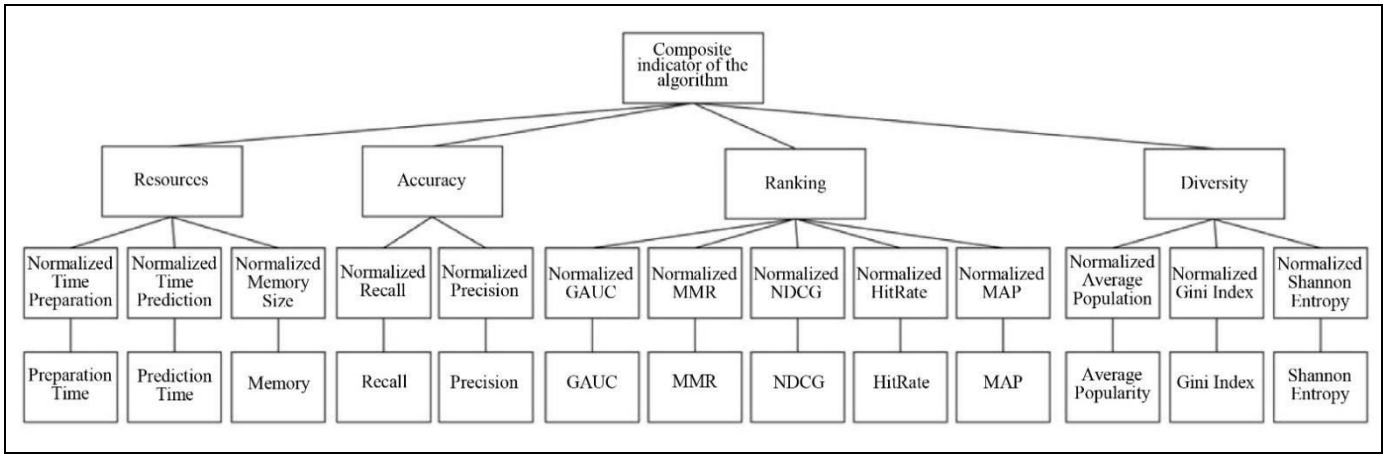

Fig. 1. The composite indicator.

Table 2

Performance of the algorithms for the MovieLens 100k dataset

Algorithm	Memory, Mb	Preparation time (T prep.), s	Prediction time (T pred.), s	Metrics									
				Recall	*Precision*	*GAUC*	*MMR*	*NDCG*	*HitRate*	*MAP*	*Average Popularity*	*Gini Index*	*Shannon Entropy*
BPR	290	20.275	0.192	0.239	0.191	0.918	0.482	0.286	0.772	0.174	241.928	0.925	0.0116
LINE	431.1	19.703	0.189	0.217	0.178	0.914	0.438	0.256	0.751	0.147	173.535	0.864	0.0104
NeuCF	454	40.751	0.923	0.238	0.189	0.919	0.459	0.277	0.766	0.165	226.602	0.911	0.0104
DMF	634.6	59.024	0.210	0.236	0.188	0.894	0.441	0.272	0.782	0.161	227.201	0.911	0.0116
SpectralCF	501.1	32.002	0.289	0.099	0.099	0.848	0.265	0.133	0.498	0.068	338.649	0.988	0.0497
LightGCN	448.3	66.221	1.191	0.226	0.177	0.909	0.453	0.268	0.750	0.161	257.067	0.945	0.0166
MultiVAE	629.6	51.722	0.532	0.237	0.182	0.911	0.463	0.276	0.766	0.165	239.964	0.923	0.0110
CDAE	516	7.853	0.220	0.098	0.096	0.698	0.279	0.135	0.504	0.071	320.434	0.986	0.0564
RaCT	404	112.653	0.662	0.263	0.201	0.918	0.486	0.298	0.808	0.178	223.918	0.892	0.0090
SLIM	416.7	0.455	0.310	0.275	0.217	0.911	0.523	0.324	0.812	0.202	246.862	0.931	0.0147
ItemKNN	521.4	0.741	0.323	0.247	0.193	0.907	0.462	0.283	0.785	0.170	217.709	0.914	0.0125
DiffRec	389.1	112.452	0.936	0.278	0.212	0.897	0.536	0.325	0.828	0.200	232.248	0.907	0.0111

Table 3

Performance of the algorithms for the MovieLens 1m dataset

Algorithm	Memory, Mb	Preparation time (T prep.), s	Prediction time (T pred.), s	Metrics									
				Recall	*Precision*	*GAUC*	*MMR*	*NDCG*	*HitRate*	*MAP*	*Average Popularity*	*Gini Index*	*Shannon Entropy*
BPR	790.5	210.782	2.695	0.163	0.200	0.927	0.445	0.256	0.742	0.152	1147.6	0.885	0.0035
LINE	931.2	190.290	2.680	0.147	0.175	0.920	0.405	0.225	0.707	0.129	721.73	0.791	0.0035
NeuCF	439.5	369.691	13.162	0.145	0.184	0.924	0.403	0.229	0.710	0.132	1136.8	0.896	0.0040
DMF	926.4	1666.525	2.468	0.152	0.188	0.888	0.424	0.239	0.724	0.138	1265.9	0.928	0.0054
SpectralCF	614.7	5327.285	2.691	0.142	0.187	0.922	0.411	0.233	0.701	0.137	1142	0.891	0.0037
LightGCN	601.2	5217.235	2.815	0.155	0.166	0.924	0.390	0.214	0.743	0.112	1136.1	0.889	0.0037
MultiVAE	1019.1	409.670	4.715	0.179	0.200	0.924	0.447	0.260	0.769	0.151	1137.6	0.876	0.0034
CDAE	1215.4	517.649	3.052	0.147	0.192	0.912	0.435	0.243	0.710	0.143	1511.7	0.954	0.0055
RaCT	1122.5	556.044	4.915	0.175	0.196	0.924	0.433	0.252	0.763	0.145	1128.3	0.875	0.0033
SLIM	916.7	6.819	5.379	0.193	0.225	0.909	0.503	0.296	0.791	0.182	1421.5	0.951	0.0068
ItemKNN	1739.1	9.702	4.583	0.163	0.199	0.917	0.447	0.256	0.742	0.152	1201.2	0.918	0.0045
DiffRec	797.2	500.809	7.739	0.200	0.232	0.917	0.511	0.303	0.806	0.186	1277.6	0.922	0.0053

Table 4

Performance of the algorithms for the Amazon Gift Card dataset

Algorithm	Memory, Mb	Preparation time (T prep.), s	Prediction time (T pred.), s	Metrics									
				Recall	*Precision*	*GAUC*	*MMR*	*NDCG*	*HitRate*	*MAP*	*Average Popularity*	*Gini Index*	*Shannon Entropy*
BPR	734.4	26.275	4.171	0.076	0.008	0.550	0.028	0.039	0.076	0.028	796.207	0.898	0.0115
LINE	1099.6	45.129	4.123	0.067	0.007	0.505	0.026	0.035	0.067	0.026	755.302	0.888	0.0091
NeuCF	1156.8	64.381	6.920	0.194	0.019	0.909	0.089	0.114	0.194	0.089	2974.811	0.993	0.1484
DMF	1268.3	319.145	4.464	0.219	0.022	0.862	0.093	0.122	0.219	0.093	2849.264	0.993	0.0993
SpectralCF	1356.6	179.778	3.587	0.213	0.021	0.908	0.092	0.121	0.213	0.092	2955.627	0.993	0.1501
LightGCN	1389.8	373.526	4.091	0.112	0.011	0.635	0.048	0.063	0.112	0.048	836.715	0.900	0.0087
MultiVAE	1515.4	152.994	5.120	0.315	0.032	0.915	0.125	0.170	0.315	0.125	1896.761	0.957	0.0061
CDAE	1505.8	48.741	4.803	0.212	0.021	0.786	0.093	0.121	0.212	0.093	2884.900	0.993	0.1631
RaCT	1514.7	191.619	5.590	0.319	0.032	0.913	0.125	0.170	0.319	0.125	1918.316	0.955	0.0061
SLIM	718.8	1.457	4.193	0.004	0.001	0.500	0.001	0.002	0.004	0.001	106.559	0.994	0.1921
ItemKNN	1146.1	3.545	3.960	0.172	0.017	0.741	0.073	0.096	0.172	0.073	316.197	0.875	0.0093
DiffRec	1408.8	231.541	11.727	0.231	0.023	0.758	0.101	0.132	0.232	0.101	2291.953	0.970	0.0096

Table 5

Normalized performance of the algorithms for the MovieLens 100k dataset

Algorithm	Memory, Mbs	Preparation time (T prep.), s	Prediction time (T pred.), s	Metrics									
				Recall	*Precision*	*GAUC*	*MMR*	*NDCG*	*HitRate*	*MAP*	*Average Popularity*	*Gini Index*	*Shannon Entropy*
BPR	1	0.8234	0.9965	0.7835	0.7926	0.9946	0.8007	0.8004	0.8296	0.7869	0.5858	0.4919	0.0549
LINE	0.5905	0.8285	1	0.6602	0.6786	0.9743	0.6364	0.6404	0.7653	0.5902	1	0	0.0295
NeuCF	0.5240	0.6409	0.2677	0.7790	0.7719	1	0.7172	0.7535	0.8102	0.7213	0.6786	0.3793	0.0295
DMF	0	0.4780	0.9785	0.7685	0.7669	0.8866	0.6501	0.7285	0.8584	0.6930	0.6750	0.3834	0.0549
SpectralCF	0.3874	0.7189	0.8995	0.0056	0.0248	0.6766	0	0	0	0	0	1	0.8587
LightGCN	0.5406	0.4138	0	0.7135	0.6711	0.9540	0.6932	0.7040	0.7620	0.6915	0.4941	0.6556	0.1603
MultiVAE	0.0145	0.5431	0.6576	0.7740	0.7141	0.9621	0.7309	0.7467	0.8102	0.7221	0.5977	0.4741	0.0422
CDAE	0.3442	0.9341	0.9684	0	0	0	0.0517	0.0135	0.0161	0.0179	0.1103	0.9854	1
RaCT	0.6692	0	0.5277	0.9167	0.8694	0.9955	0.8136	0.8629	0.9391	0.8204	0.6949	0.2285	0
SLIM	0.6323	1	0.8791	0.9845	1	0.9612	0.9524	0.9953	0.9518	1	0.5559	0.5397	0.1203
ItemKNN	0.3285	0.9975	0.8656	0.8290	0.8066	0.9435	0.7279	0.7858	0.8681	0.7563	0.7325	0.4003	0.0738
DiffRec	0.7124	0.0018	0.2549	1	0.9587	0.8975	1	1	1	0.9821	0.6444	0.3477	0.0443

For all evaluation criteria, the standard deviation is calculated on each dataset:

$$f_j = \sqrt{\frac{\sum_{i=1}^{N}(e_{ij} - \overline{e_j})^2}{N-1}},$$

where e_{ij} denotes the normalized value; $\overline{e_j}$ is the mean value of the metric for all algorithms; N is the number of algorithms.

Table 6 presents the standard deviations of different parameters.

Next, we determine the weights in each layer within each group of partial criteria and subindices (v_j denote the weights in the first layer). To calculate the weights, the standard deviation of a particular parameter is divided by the sum of the standard deviations in the group:

$$v_j = \frac{f_j}{\sum_{j=1}^{N_p} f_j},$$

where N_p denotes the number of metrics in the pth group.

Then, the value of each pth subindicator is found for all algorithms in the first layer via additive convolution:

$$a_{ip} = \sum_{j=1}^{N_p} e_{ij} v_j .$$

The next step is to calculate the standard deviations in the subindices:

$$s_p = \sqrt{\frac{\sum_{i=1}^{N}(a_{ip} - \overline{a_p})^2}{N-1}},$$

Table 6

Standard deviations for the MovieLens 100k dataset

Metric	Standard deviation
Memory	0.2198
T prep.	0.2730
T pred.	0.2914
Recall	0.2313
Precision	0.2196
GAUC	0.1718
MMR	0.2092
NDCG	0.2256
HitRate	0.2365
MAP	0.2229
Average Popularity	0.1827
Gini Index	0.2034
Shannon Entropy	0.2412

where $\overline{a}_p$ denotes the mean value of the pth subindicator.

Similar to the first layer, we calculate the weights in the second layer:

$$w_p = \frac{s_p}{\sum_{p=1}^{P} s_p},$$

where P is the number of subindices.

Table 7 shows the distribution of the metrics across groups, the sum of standard deviations in the group, and the weights of the first and second layers.

Table 7

Weight calculation for the MovieLens 100k dataset

Metric	Group	Sum of standard deviations in the group	Weight of the first layer	Weight of the second layer
Memory	1	0.784	0.280	0.274
T prep.	1		0.348	
T pred.	1		0.371	
Recall	2	0.451	0.512	0.303
Precision	2		0.487	
GAUC	3	1.066	0.161	0.286
MMR	3		0.196	
NDCG	3		0.211	
HitRate	3		0.221	
MAP	3		0.209	
Average Popularity	4	0.627	0.291	0.135
Gini Index	4		0.324	
Shannon Entropy	4		0.384	

The block diagrams in Figs. 2–4 demonstrate the composite indicator model with the weights of each parameter for the MovieLens 100k, MovieLens 1m, and Amazon Gift datasets, respectively.

Passing to the subindicators implies calculating the normalized values. For ease of understanding, Figs. 2–4 do not include this network layer, despite its factual presence, as in Fig. 1.

Table 8 shows the values calculated in the first layer. They characterize the evaluation for each group of the criteria (Resources, Accuracy, Ranking, and Diversity).

The last step is to perform the additive convolution of the second layer by analogy to the first layer:

$$b_i = \sum_{p=1}^{P} a_{ip} w_p ,$$

where P denotes the number of values in the first layer; the index i corresponds to the algorithm for which the calculations are carried out.

Table 9 presents the final value of the composite indicator for each dataset and the arithmetic mean of the evaluations of different datasets. The values in the cells characterize the integral evaluation of the algo-

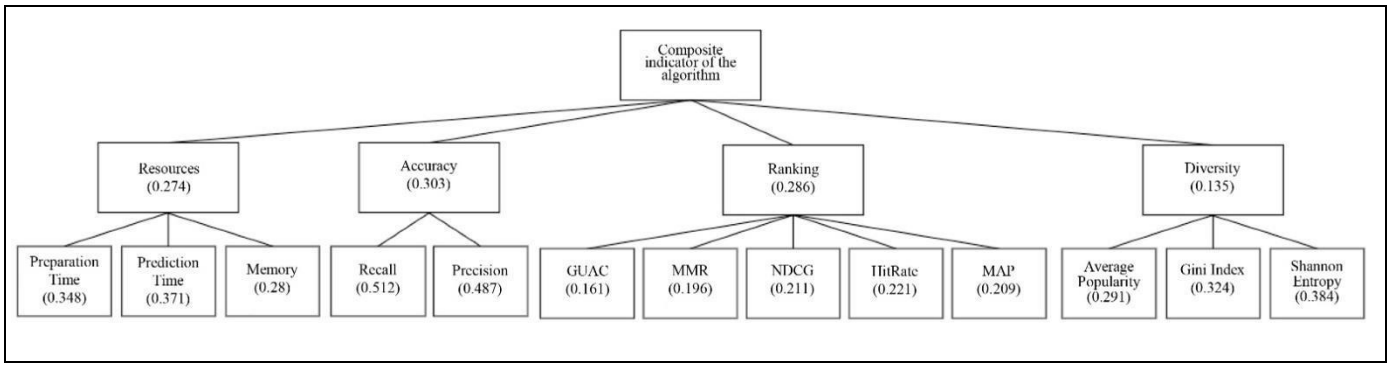

Fig. 2. The composite indicator weights for the MovieLens 100k dataset.

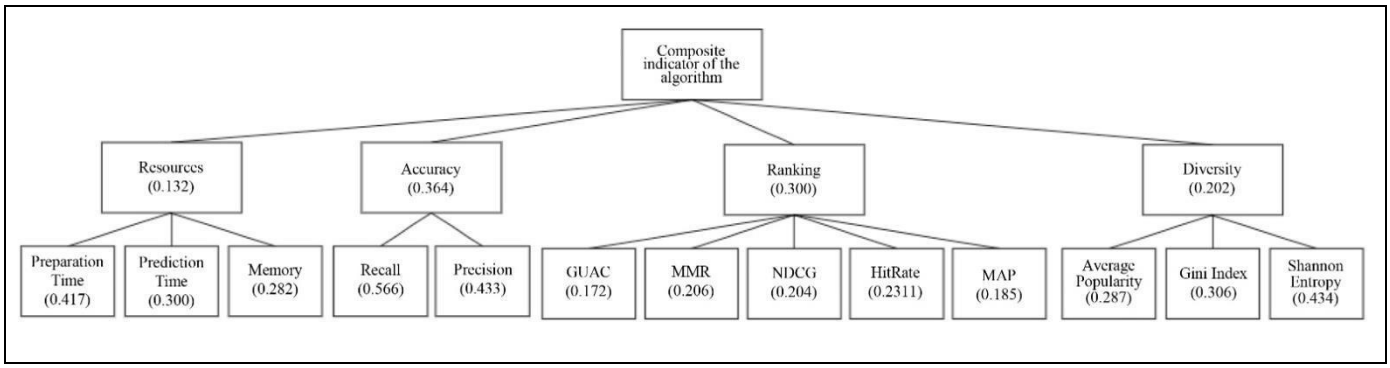

Fig. 3. The composite indicator weights for the MovieLens 1m dataset.

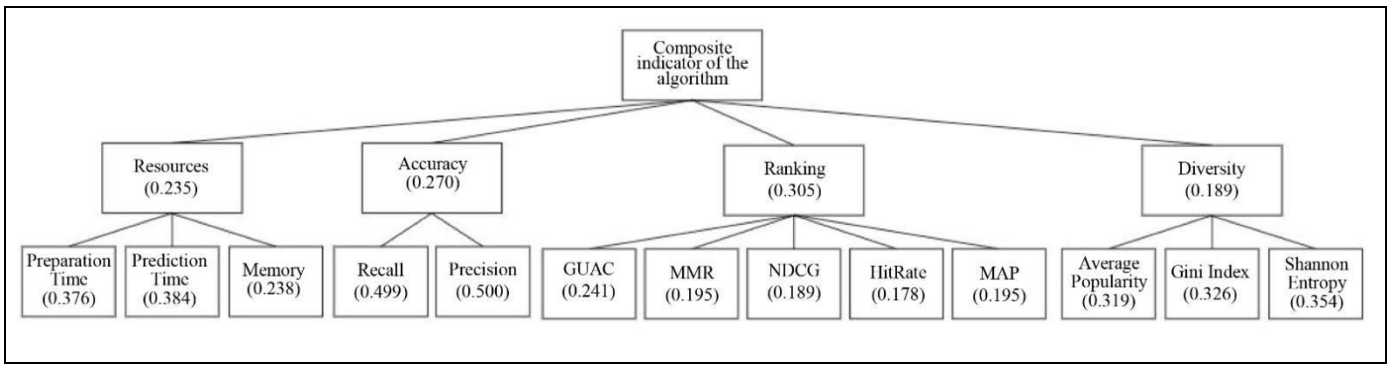

Fig. 4. The composite indicator weights for the Amazon Gift Card dataset.

Table 8

Calculation of subindicator values in the first layer for the MovieLens 100k dataset

Algorithm	Group / Subindicator			
	Resources	Accuracy	Ranking	Diversity
BPR	0.9372	0.7879	0.8354	0.3512
LINE	0.8255	0.6691	0.7106	0.3026
NeuCF	0.4695	0.7755	0.7920	0.3320
DMF	0.5300	0.7677	0.7600	0.3419
SpectralCF	0.6931	0.0149	0.1091	0.6544
LightGCN	0.2956	0.6928	0.7524	0.4181
MultiVAE	0.4374	0.7448	0.7873	0.3440
CDAE	0.7814	0	0.0203	0.7361
RaCT	0.3836	0.8936	0.8826	0.2765
SLIM	0.8520	0.9920	0.9727	0.3831
ItemKNN	0.7610	0.8181	0.8120	0.3715
DiffRec	0.2950	0.9799	0.9798	0.3174

Table 9

Final evaluation

Algorithm	MovieLens 100k	MovieLens 1m	Amazon Gift Card	Arithmetic mean of the evaluations
SLIM	0.8656	0.8390	0.4202	0.7083
DiffRec	0.7022	0.8649	0.5328	0.7000
MultiVAE	0.6184	0.5620	0.7356	0.6387
RaCT	0.6670	0.5058	0.7253	0.6327
ItemKNN	0.7402	0.4963	0.5591	0.5985
BPR	0.7834	0.5054	0.4051	0.5646
DMF	0.6426	0.3799	0.6043	0.5423
NeuCF	0.6362	0.3123	0.6525	0.5337
CDAE	0.3199	0.4090	0.6428	0.4572
LINE	0.6743	0.2874	0.3340	0.4319
SpectralCF	0.3145	0.2811	0.6506	0.4154
LightGCN	0.5637	0.2664	0.3265	0.3855

rithm by the set of criteria. The algorithms in this table are in descending order of the arithmetic mean of the evaluations.

According to Table 9, the same algorithms have different evaluations on the MovieLens 100k and 1m datasets, depending on the dataset size. For example, LINE demonstrates a high evaluation for the first dataset, but it drops by more than a factor of two for the second dataset. For SLIM, the evaluation remains almost the same when increasing the dataset size. Some algorithms (DiffRec, CDAE) show an increase in the evaluation. Thus, there is no obvious dependence of the algorithm's evaluation on the dataset size. This conclusion is also confirmed by the Pearson correlation coefficient between the algorithms on the MovieLens 100k and 1m datasets: the value is only 0.56, corresponding to a rather weak dependence.

The algorithms' evaluations on the MovieLens and Amazon Gift Card datasets do not correlate with each other. For example, BPR shows a high evaluation on the MovieLens dataset and a low evaluation on the Amazon Gift Card dataset.

A deeper analysis of the table with the algorithms' evaluations confirms the initial hypothesis: the same algorithm behaves differently and gives different performance depending on the dataset.

There is no clearly defined reason for this phenomenon, as each group of algorithms involves different principles. For example, clustering-based algorithms may inaccurately predict the element class due to

fuzzy cluster bounds if there are insufficient data. At the same time, under enough data and complete cluster bounds, the clustering results are accurate. Similar problems may arise in algorithms based on other principles.

Also, it is necessary to tune the hyperparameters of algorithms according to the data context considering more than one characteristic metric. A single metric can be applied to solve this problem.

4. DISCUSSION

The results of this study indicate the practical applicability of the proposed approach for selecting an optimal recommendation algorithm, especially in the context of particular datasets. Note that the method can be further improved by introducing user ranking to consider individual user preferences and prioritize the criteria most significant for them. For example, a user appreciating the accuracy of recommendations higher than the algorithm speed can assign appropriate weights, making the approach more flexible and personalized.

In addition, this approach can be applied to optimize the algorithms' hyperparameters. As a result, it becomes possible to systematically estimate the variations of hyperparameters and their impact on particular metrics and the total algorithm evaluation. The systematic estimation of the hyperparameters using this approach allows implementing an iterative model improvement process. Developers will be able to regularly analyze and optimize the parameters in accordance with changing requirements and data.

Hyperparameter optimization improves algorithm performance and accuracy. It also shows which parameters are most significant for achieving the best results: different tasks may require different optimal parameter settings.

An important line is the possible application of this method to form ensembles of algorithms for recommender systems. Evaluation maximization allows effectively combining different algorithms into an ensemble considering their contribution to the recommender system. Such an approach allows developing a more robust and effective recommender system that will cover the advantages and disadvantages of each algorithm to mitigate the shortcomings of particular algorithms and generate more accurate and relevant recommendations for users.

The approaches presented above have considered only the offline evaluations of recommendation algorithms, perhaps forming an incomplete picture of their effectiveness. In the future, a composite indicator may be developed to provide the online evaluation of such algorithms. It is also possible to combine online and offline metrics into a generalized measure. They can be based on such metrics as *Gross merchandise volume* (*GMV*), *Click-through rate* (*CTR*), and *Conversion rate* (*CVR*). These metrics will integrally evaluate the impact of recommendation algorithms on business indicators considering their offline and online aspects.

Other research areas may include analyzing the impact of dynamic data changes on the performance of recommendation algorithms. It is possible to study the performance of a method under changing user preferences, seasonal fluctuations, or temporal trends. Investigating the impact of changing trends in consumer behavior may be a key step toward developing more adaptive and stable recommender systems.

CONCLUSIONS

This paper has considered the problem of generating a composite indicator for evaluating the performance of recommender systems. For this purpose, different algorithms and datasets have been selected, including MovieLens and Amazon Gift Card. For each algorithm and dataset, different metrics have been calculated to represent different aspects of recommendation quality and reflect their accuracy, completeness, diversity, and other characteristics. The composite indicator has been formed by combining these metrics using the entropy-based weighting method. This approach considers the relative significance of each metric and provides balance in the comprehensive evaluation of the effectiveness of recommender systems.

The application of such an approach appears to be novel in recommender system evaluation. The entropy-based composite indicator is a unique tool for comparing different algorithms and datasets in terms of their overall performance. This approach covers different aspects of recommendation quality and can be applied in different recommender system scenarios.

According to the results presented, the composite indicator can be a useful tool for evaluating the effectiveness of recommender systems by combining various metrics into a single overall evaluation. Hence, it gains superiority over partial criteria in terms of informativeness and usability. This indicator provides a deeper and more comprehensive understanding of the effectiveness of recommender systems, representing a valuable tool for making well-grounded decisions in this area.

However, forming the composite indicator requires a large amount of calculations and data analysis. In addition, the choice of algorithms and datasets may significantly affect the evaluation results. Therefore, additional research and experiments with different al-

gorithms and datasets should be carried out to obtain more accurate results. These studies will provide a better understanding of the impact of different parameters on the final results and optimize the composite indicator formation process. With this approach, one will assess how well the composite indicator adapts to different recommender system scenarios and identify its limitations.

Additional research may also optimize the weights of combining different metrics into the composite indicator. This is crucial to consider the relative significance of each metric and balance the different aspects of recommendation quality.

Thus, further research on forming composite indicators for evaluating the effectiveness of recommender systems is necessary to improve their accuracy, generalizability, and applicability.

Acknowledgments. *This work was carried out within the state order of the Ministry of Science and Higher Education of the Russian Federation, project no. FEWM-2023-0013.*

REFERENCES

1. Fayyaz, Z., Ebrahimian, M., Nawara, D., et al., Recommendation Systems: Algorithms, Challenges, Metrics, and Business Opportunities, *Appl. Sci.*, 2020, vol. 10, no. 21, art. no. 7748. DOI: https://doi.org/10.3390/app10217748.
2. Amatriain, X. and Basilico, J., Recommender Systems in Industry: A Netflix Case Study, in *Recommender Systems Handbook*, Ricci, F., Rokach, and Shapira, B., Eds., Boston, MA: Springer, 2015, pp. 385–419. DOI: https://doi.org/10.1007/978-1-4899-7637-6_11.
3. Wibisono, C., Purwanti, E., and Effendy, F., A Systematic Literature Review of Movie Recommender Systems for Movie Streaming Service, *AIP Conf. Proc.*, 2023, vol. 2554, no. 1, art. no. 040005. DOI: https://doi.org/10.1063/5.0104316.
4. Kulshin, R., Sidorov, A., and Senchenko, P. Using Neural Networks with Reinforcement in the Tasks of Forming User Recommendations, *Journal of Physics: Conference Series*, 2022, vol. 2291, no. 1, art. no. 012005. DOI: 10.1088/1742-6596/2291/1/012005.
5. Sipser, M., *Introduction to the Theory of Computation. Course Technology*. Boston, MA: PWS Publishing, 2006.
6. Aixin, S., On Challenges of Evaluating Recommender Systems in an Offline Setting, *Proceedings of the* 17th ACM Conference on Recommender Systems, Singapore, 2023, pp. 1284–1285.
7. Jimenez-Fernandez, E. and Ruiz-Martos, M., Review of Some Statistical Methods for Constructing Composite Indicators, *Studies of Applied Economics*, 2020, vol. 38, no. 1, pp. 1–15. DOI: https://doi.org/10.25115/eea.v38i1.3002.
8. Nardo, M., Saisana, M., Saltelli, A., and Tarantola, S., Tools for Composite Indicators Building, *Report no. JRC31473*, Rome, Italy: European Commission, ISPRA, 2005.
9. Becker, W., Saisana, M., Paruolo, P., and Vandecasteele, I., Weights and Importance in Composite Indicators: Closing the Gap, *Ecological Indicators*, 2017, vol. 80, pp. 12–22. DOI: https://doi.org/10.1016/j.ecolind.2017.03.056.
10. Sidorov, A.A., Methodological Approach to the Integral Assessment of the State and Dynamics of Multidimensional Objects of Socio-economic Nature, *Control Sciences*, 2016, no. 3, pp. 32–40. (In Russian.)
11. Abberger, K., Graff, M., Müller, O., and Sturm, J., Composite Global Indicators from Survey Data: The Global Economic Barometers, *Rev. World Econ.*, 2022, vol. 158, pp. 917–945. DOI: https://doi.org/10.1007/s10290-021-00449-8.
12. Endrodi-Kovacs, V. and Tankovsky, O., A Composite Indicator for Economic Integration Maturity: The Case of Western Balkan Countries, *Eastern Journal of European Studies*, 2022, vol. 13, no 1, pp. 148–166. DOI: 10.47743/ejes-2022-0107.
13. Khadzhynova, O., Simanaviciene, Z., Mints, O., et al., Assessment of the EU Countries' Economic Security Based on the Composite Indicators, *WSEAS Transactions on Business and Economics*, 2022, vol. 19, pp. 690–700. DOI: 10.37394/23207.2022.19.61.
14. McDonnell, T., Cosgrove, G., Hogan, E., et al., Methods to Derive Composite Indicators Used for Quality and Safety Measurement and Monitoring in Healthcare: A Scoping Review Protocol, *BMJ Open.*, 2023, vol. 13, no. 7. DOI: 10.1136/bmjopen-2022-071382.
15. Kara, P., Valentin, J., Mainz, J., and Johnsen, S., Composite Measures of Quality of Health Care: Evidence Mapping of Methodology and Reporting, *PLoS One*, 2022, vol. 17, no. 5. DOI: 10.1371/journal.pone.0268320.
16. Asadi-Lari, M., Majdzadeh, R., Mansournia, M.A, et al., Construction and Validation of CAPSES Scale as a Composite Indicator of SES for Health Research: An Application to Modeling Social Determinants of Cardiovascular Diseases, *BMC Public Health.*, 2023, vol. 23, art. no. 293. DOI: https://doi.org/10.1186/s12889-023-15206-9.
17. Abenayake, C., Mikami, Y., Matsuda, Y., and Jayasinghe, A., Ecosystem Services-Based Composite Indicator for Assessing Community Resilience to Floods, *Environmental Development*, 2018, vol. 27, pp. 34–46. DOI: https://doi.org/10.1016/j.envdev.2018.08.002.
18. Gómez-Limón, J., Arriaza, M., and Guerrero-Baena, M., Building a Composite Indicator to Measure Environmental Sustainability Using Alternative Weighting Methods, *Sustainability*, 2020, vol. 12, no. 11, art. no. 4398. DOI: https://doi.org/10.3390/su12114398.
19. Alam, M., Dupras, J., and Messier, C., A Framework towards a Composite Indicator for Urban Ecosystem Services, *Ecological Indicators*, 2016, vol. 60, pp. 38–44. DOI: https://doi.org/10.1016/j.ecolind.2015.05.035.
20. Melo-Aguilar, C., Agulles, M., and Jordà, G., Introducing Uncertainties in Composite Indicators. The Case of the Impact Chain Risk Assessment Framework, *Front. Clim.*, 2022, vol. 4. DOI: https://doi.org/10.3389/fclim.2022.1019888.
21. Dolge, K. and Blumberga, D., Composite Risk Index for Designing Smart Climate and Energy Policies, *Environmental and Sustainability Indicators*, 2021, vol. 12, art. no. 100159. DOI: https://doi.org/10.1016/j.indic.2021.100159.
22. Do, H., Ly, T., and Do, T., Combining Semi-Quantitative Risk Assessment, Composite Indicator and Fuzzy Logic for Evaluation of Hazardous Chemical Accidents, *Sci. Rep.*, 2020, vol. 10, art. no. 18544. DOI: https://doi.org/10.1038/s41598-020-75583-8.
23. Avanesian, G., Mizunoya, S., and Delamonica, E., UNICEF Remote Learning Readiness Index: A Composite Indicator to Assess Resilience of Education Sector against Crises and

Emergencies, *Statistical Journal of the IAOS*, 2022, vol. 38, pp. 1–14. DOI: 10.3233/SJI-220051.
24. Segovia-Gonzalez, M. and Contreras, I., A Composite Indicator to Compare the Performance of Male and Female Students in Educational Systems, *Soc. Indic. Res.*, 2023, vol. 165, pp. 181–212. DOI: https://doi.org/10.1007/s11205-022-03009-1.
25. Hubelova, D., Odvarkova, V., and Chalupa, P., Selected Factors of Education Level in East African Countries: Comparative Method Using Composite Indicator, *Geographical Journal*, 2016, vol. 68, pp. 55–72.
26. Silveira, T., Zhang, M., Lin, X., et al., How Good Your Recommender System Is? A Survey on Evaluations in Recommendation, *Journal of Machine Learning and Cybernetics*, 2016, vol. 10, pp. 813–831. DOI: https://doi.org/10.1007/s13042-017-0762-9.
27. Hongzhi, Y., Cui, B., Li, J., et al., Challenging the Long Tail Recommendation, *Proceedings of the VLDB Endowment*, 2012, vol. 5, pp. 896–907. DOI: https://doi.org/10.14778/2311906.2311916.
28. Hanczar, B., Hua, J., Sima, C., et al., Small-Sample Precision of ROC-Related Estimates, *Bioinformatics*, 2010, vol. 26, pp. 822–830. DOI: https://doi.org/10.1093/bioinformatics/btq037.
29. Calders, T. and Jaroszewicz, S., Efficient AUC Optimization for Classification, in *Lecture Notes in Computer Science*, 2007, vol. 4702, pp. 42–53. DOI: https://doi.org/10.1007/978-3-540-74976-9_8.
30. Wenlong, S., Khenissi, S., Nasraoui, O., and Shafto, P., Debiasing the Human-Recommender System Feedback Loop in Collaborative Filtering, *Proceedings of the 2019 World Wide Web Conference*, San Francisco, 2019, pp. 645–651. DOI: https://doi.org/10.1145/3308560.3317303.
31. Zhang, Q., Cao, L., Zhu, C., et al., CoupledCF: Learning Explicit and Implicit User-item Couplings in Recommendation for Deep Collaborative Filtering, *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, Stockholm, 2018, pp. 3662–3668. DOI: https://doi.org/10.24963/ijcai.2018/509.
32. Bellogin, A., Castells, P., and Cantador, I., Precision-Oriented Evaluation of Recommender Systems: An Algorithmic Comparison, *Proceedings of the 5th ACM Conference on Recommender Systems*, Chicago, 2011, pp. 333–336. DOI: https://doi.org/10.1145/2043932.2043996.
33. Wang, Y., Application of Recall Methods in Recommendation Systems, *Proceedings of the 3rd International Conference on Signal Processing and Machine Learning*, Oxford, 2023, vol. 4, pp. 44–51. DOI 10.54254/2755-2721/4/20230344.
34. Zriaa, R., Sadiki, H., Ertel, M., et al., Qualitative Recommender System Using Entropy-Weighted Pedagogical Criteria for Effective Training in E-Learning Platforms, *Journal of Theoretical and Applied Information Technology*, 2023, vol. 101, no. 9, pp. 3517–3529.
35. Kumar, C. and Kumar, M., User Session Interaction-Based Recommendation System Using Various Machine Learning Techniques, *Multimed. Tools Appl.*, 2023, vol. 82, pp. 21279–21309.
36. Wang, Y., Wang, L, Li, Y., et al., A Theoretical Analysis of NDCG Ranking Measures, *Proceedings of the 26th Annual Conference on Learning Theory*, Princeton, 2013, vol. 30, pp. 25–54.
37. Steffen, R., Freudenthaler, C., Gantner, Z., and Schmidt, L., BPR: Bayesian Personalized Ranking from Implicit Feedback, *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, Montreal, 2009, pp. 452–461.
38. Jian, T., Qu, M., Wang, M., et al., LINE: Large-Scale Information Network Embedding, *Proceedings of the 24th International Conference on World Wide Web*, Florence, 2015, pp. 1067–1077. DOI: https://doi.org/10.1145/2736277.2741093.
39. He, X., Liao, L., Zhang, H., et al., Neural Collaborative Filtering, *Proceedings of the 26th International Conference on World Wide Web*, Perth, 2017, pp. 173–182. DOI: https://doi.org/10.1145/3038912.3052569.
40. Xue, J, Dai, X., Zhang, J., et al., Deep Matrix Factorization Models for Recommender Systems, *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, Toronto, 2017, pp. 3203–3209. DOI: https://doi.org/10.24963/ijcai.2017/447.
41. Lei, Z., Lu, C., Jiang, F., et al., Spectral Collaborative Filtering, *Proceedings of the 12th ACM Conference on Recommender Systems*, Vancouver, 2018, pp. 311–319. DOI: https://doi.org/10.1145/3240323.3240343.
42. He, X., Deng, K., Wang, X., et al., LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation, *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Xian, 2020, pp. 639–648. DOI: https://doi.org/10.1145/3397271.3401063.
43. Dawen, L., Krishnan, R., Hoffman, M., and Jebara, T., Variational Autoencoders for Collaborative Filtering, *Proceedings of the 2018 World Wide Web Conference*, Lyon, 2018, pp. 689–698. DOI: https://doi.org/10.1145/3178876.3186150.
44. Yao, W., DuBois, C., Zheng, A., and Ester, M., Collaborative Denoising Auto-Encoders for Top-N Recommender Systems, *Proceedings of the 9th ACM International Conference on Web Search and Data Mining*, San Francisco, 2016, pp. 153–162. DOI: https://doi.org/10.1145/2835776.2835837.
45. Lobel, S., Li, C., Gao, J., and Carin, L., RaCT: Toward Amortized Ranking-Critical Training for Collaborative Filtering, *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, Addis Ababa, 2020.
46. Ning, X. and Karypis, G., SLIM: Sparse Linear Methods for Top-N Recommender Systems, *Proceedings of the IEEE 11th International Conference on Data Mining*, Vancouver, 2011, pp. 497–506. DOI: 10.1109/ICDM.2011.134.
47. Mukund, D. and Karypis, G., Item-Based Top-N Recommendation Algorithms, *ACM Transactions on Information Systems*, 2004, vol. 22, pp. 143–177. DOI: https://doi.org/10.1145/963770.963776.
48. Wang, W., Xu, Y., Feng, F., et al., Diffusion Recommender Model, *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Taipei, 2023, pp. 832–841. DOI: https://doi.org/10.1145/3539618.3591663.
49. Zhao, W., Mu, S., Hou, Y., et al., RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms, *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, Queensland, 2021, pp. 4653–4664. DOI: https://doi.org/10.1145/3459637.3482016.
50. Raev, A.G., On One Way to Determine Weighting Coefficients of Particular Criteria in Development of an Integral Additive Criterion, *Avtomat. Telemekh.*, 1984, no. 5, pp. 162–165. (In Russian.)

This paper was recommended for publication by RAS Academician D.A. Novikov, a member of the Editorial Board.

Received March 5, 2024, and revised May 14, 2024. Accepted July 19, 2024.

Author information

Kulshin, Roman Sergeevich. Postgraduate, Tomsk State University of Control Systems and Radioelectronics, Tomsk, Russia
✉ roman.s.kulshin@tusur.ru
ORCID iD: https://orcid.org/0000-0002-6891-1869

Sidorov, Anatolii Anatol'evich. Cand. Sci. (Econ.), Tomsk State University of Control Systems and Radioelectronics, Tomsk, Russia
✉ anatolii.a.sidorov@tusur.ru
ORCID iD: https://orcid.org/0000-0002-9236-3639

Cite this paper

Kulshin, R.S. and Sidorov, A.A., An Entropy-Based Composite Indicator for Evaluating the Effectiveness of Recommender System Algorithms. *Control Sciences* **4**, 37–51 (2024). http://doi.org/10.25728/cs.2024.4.4

Original Russian Text © Kulshin, R.S., Sidorov, A.A., 2024, published in *Problemy Upravleniya*, 2024, no. 4, pp. 44–60.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences,
Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com

DOI: http://doi.org/10.25728/cs.2024.4.5

AN IDENTIFICATION-BASED CONTROL METHOD FOR AN OVERHEAD CRANE WITH A NEW COMBINED SENSOR PLACEMENT

S. P. Kruglov*, S. V. Kovyrshin**, and D. V. Butorin***

Irkutsk State Transport University, Irkutsk, Russia

*✉ kruglov_s_p@mail.ru, **✉ sergkow@mail.ru, ***✉ den_butorin@mail.ru

Abstract. This paper is devoted to an automatic control method for an overhead crane under the current parametric uncertainty of the crane, transported cargo, and exogenous disturbances. The control objective is to move the cargo in the horizontal plane to a point ensuring the final delivery of the cargo to the designated place while parrying the angular oscillations of the suspension and reaching given dynamic characteristics. The approach is based on a control scheme with a current parametric identification algorithm, an implicit reference model, and "simplified" adaptability conditions to track cargo movements directly. The control law generates a given trolley speed for the servo drive. The passport data of the crane installation are used to select the control law parameters. Unlike previous publications on the topic, the solution proposed below is simpler, more reliable in terms of operation, and less expensive. This is achieved by placing a combined sensor (an accelerometer with an angular rate sensor (ARS)) on a suspension cable near the crane trolley and applying, first, an algorithmic solution without the preliminary calculation of the ARS drift and, second, a current parametric identification procedure of higher efficiency. Computer simulation results are provided to confirm these advantages of the new solution. A similar example is implemented on an experimental installation.

Keywords: overhead crane, control automation, current parametric uncertainty, parametric identification algorithm, approximation.

INTRODUCTION

During the operation of different types of cranes, including overhead ones, the automatic transfer of suspended cargos is a topical problem. This is due to the high prevalence of such cranes and the need to damp pendulum oscillations, which reduce safety, productivity, the accuracy of work, etc. It is especially important to implement automation under the current parametric uncertainty of the crane installation, transported cargo, and exogenous disturbances, caused by a wide variety of crane operating modes.

There exist many automatic control methods for overhead cranes, which solve this problem in different formulations:

- approaches based on PID, PI, and PD controllers (for example, see [1–3]). These methods suffer from the drawback that the controllers have to be pre-tuned for a particular range of crane and load parameters.
- sliding mode control methods for overhead cranes (for example, see [4, 5]). Their main shortcoming is the appearance of high-frequency components in drive control, which worsens the performance characteristics of the control system.
- neural network controllers for overhead cranes [6, 7]. They are disadvantageous due to a considerable training time of the neural network.
- an adaptive control method for an overhead crane trolley that involves a Lyapunov function with the tuning of controller parameters based on the gradient identification algorithm [8, 9]. The main disadvantage of this method is the problem of selecting the parameters of the discrete identification algorithm for a particular case to make the closed-loop control system stable.

Another control approach for an overhead crane under current parametric uncertainty was considered in [10, 11]. It is based on an adaptive control scheme

with a parametric identifier, an implicit reference model, and "simplified" adaptability conditions. It was proposed to use an asynchronous servo drive, which is fast enough in the full range of working loads [12]. The control law generates a given speed for the actuators to track linear cargo movements directly. This paper is a logical continuation of the studies mentioned to simplify the control system design and reduce the operating costs of an overhead crane by improving the control approach.

1. PROBLEM STATEMENT

We will consider the motion of the crane and the cargo only along one axis (the horizontal movements of the cargo) corresponding to the movements of the crane trolley; see the diagram in Fig. 1. (The other axis corresponding to the movements of the crane beam can be treated by analogy.)

Figure 1 has the following notations:

m_T and m_c are the masses of the crane trolley and transported cargo, respectively, considering the inertia of the rotating masses (the cargo consists of the load transferred and the gripping device, i.e., a hook);

r_c is the radius of the cargo inertia;

l is the length of the cargo suspension (the distance between the attachment point of the suspension on the trolley and the cargo's center of gravity); by assumption, $l_{\min} \le l \le l_{\max}$, where $l_{\min}$ and $l_{\max}$ are the minimum and maximum values of the suspension length, respectively;

x_T is the horizontal movement of the trolley along the Ox axis;

$v = \dot{x}_T$ is the trolley speed with the restriction $|v| \le v_{\max}$, and v_{giv} is a given value of this speed;

f_{con} is the control force generated by the servo drive of the crane trolley using the signal v_{giv};

f_{fri} is the friction force counteracting trolley movements;

f_w is the wind force applied to the center of the cargo mass (the wind load);

φ is the deviation angle of the cargo suspension from the vertical axis;

$x_c = x_T + l\sin\varphi$ is the horizontal movement of the cargo;

S is the location point of a combined sensor (an accelerometer and an angular rate sensor (ARS)) on the suspension cable (see below);

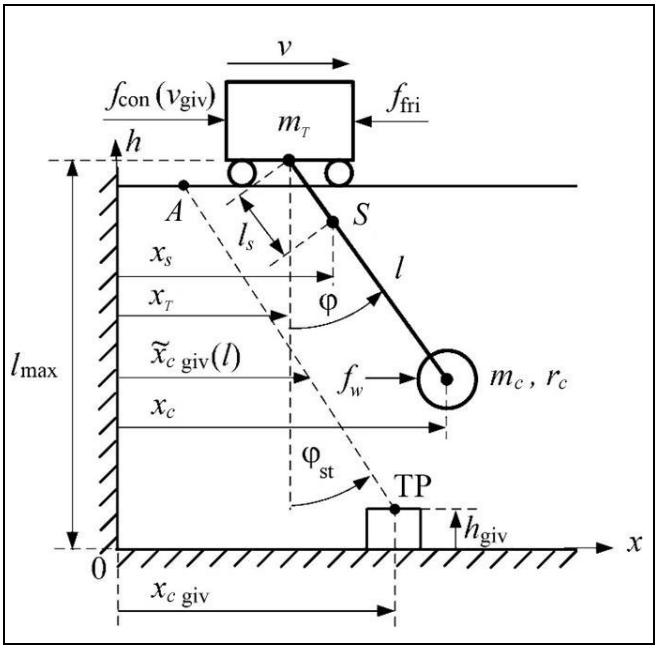

Fig. 1. Overhead crane movements with a cargo along the horizontal axis corresponding to crane trolley movements.

l_s is the suspension length to the combined sensor (the distance between the attachment point of the suspension on the trolley and point S), $0 < l_s < l_{\min}$

$x_S = x_T + l_s\sin\varphi$ is the horizontal movement of the combined sensor;

TP is the target point of cargo delivery with coordinates $\left(x_{c\,\text{giv}},\ h_{\text{giv}}\right)$, where $x_{c\,\text{giv}}$ and h_{giv} (loading height) are its coordinates along the horizontal Ox and vertical Oh axes, respectively;

$\varphi_{\text{st}} = \text{arctg}\left(f_w/(m_c g)\right)$ is the steady-state constant value of the angle φ expected at the end of the control process, where g is the free fall acceleration;

$\tilde{x}_{c\,\text{giv}}(l) = x_{c\,\text{giv}} - \left(l_{\max} - h_{\text{giv}} - l\right)\sin\varphi_{\text{st}}$ is a given position of the cargo on the horizontal axis at the current suspension length (the distance between the zero point and the A–TP line in the Ox axis.

Assume that the exogenous disturbance formed by the friction force and wind load is a step signal with limited amplitude.

The mathematical model of the mechanical system (Fig. 1) was described in [13], with the cable mass and angular motion friction neglected due to their smallness. Based on this model and the initial values of the variables characterizing position and velocity, by applying Poinsot's theorem (to include the wind effect), we obtain the following differential equations for the dynamics of this mechanical system:

$$\begin{cases} (m_T + m_c)\ddot{x}_T + (m_c l\cos\varphi)\ddot{\varphi} \\ \quad = f_{\text{con}} - f_{\text{fri}} + f_w + m_c l\dot{\varphi}^2\sin\varphi \\ (m_c l\cos\varphi)\ddot{x}_T + m_c\left(l^2 + r_c^2\right)\ddot{\varphi} \\ \quad = -m_c gl\sin\varphi + lf_w\cos\varphi \\ x_c = x_T + l\sin\varphi. \end{cases} \quad (1)$$

Usually, the angle φ has a small range (not exceeds $10-20^{\circ}$), and its rate of change is also low. In view of the motion kinematics, we can take $\sin\varphi \approx \varphi$, $\cos\varphi \approx 1$, and $\dot{\varphi}^2\sin\varphi \cong 0$. Then system (1) can be linearized to

$$\begin{cases} \ddot{x}_T \approx a_x^{f_{\text{con}}} f_{\text{con}} + a_x^{\varphi}\varphi + a_x \\ \ddot{\varphi} \approx a_{\varphi}^{f_{\text{con}}} f_{\text{con}} + a_{\varphi}^{\varphi}\varphi + a_{\varphi} \\ x_c \approx x_T + l\varphi,\ \dot{x}_c \approx \dot{x}_T + l\dot{\varphi},\ \ddot{x}_c \approx \ddot{x}_T + l\ddot{\varphi}, \end{cases} \quad (2)$$

where $a_x^{f_{\text{con}}} = \gamma^{-1} m_c\left(l^2 + r_c^2\right)$, $a_x^{\varphi} = \gamma^{-1} g\left(m_c l\right)^2$, $a_x = \gamma^{-1} m_c\left[-\left(l^2 + r_c^2\right) f_{\text{fri}} + l^2 f_w\right]$, $a_{\varphi}^{f_{\text{con}}} = -\gamma^{-1} m_c l$, $a_{\varphi}^{\varphi} = -\gamma^{-1}\left(m_T + m_c\right) m_c gl$, $a_{\varphi} = \gamma^{-1} l\left[m_c f_{\text{fri}} + \left(m_T + m_c\right) f_w\right]$, and $\gamma = m_c\left[m_T l^2 + \left(m_T + m_c\right) r_c^2\right]$.

We rewrite the third equation of the third line of system (2) using its first two lines and substitute the signal f_{con} expressed from the first equality of (2) into the resulting formula. Consequently, the cargo motion is described through the trolley speed as follows:

$$\ddot{x}_c \approx a_c^{\dot{v}}\dot{v} + a_c^{\varphi}\varphi + a_c, \quad (3)$$

where $a_c^{\dot{v}} = r_c^2 \big/ \left(r_c^2 + l^2\right)$; $a_c^{\varphi} = l\left(a_{\varphi}^{\varphi} - a_x^{\varphi} a_{\varphi}^{f_{\text{con}}} \big/ a_x^{f_{\text{con}}}\right) = -g\mu$; $\mu = l^2 \big/ \left(l^2 + r_c^2\right)$ is the dimensionless coefficient of the impact of the cargo's radius of inertia; $a_c = l\left(a_{\varphi} - a_x a_{\varphi}^{f_{\text{con}}} \big/ a_x^{f_{\text{con}}}\right)$ is the value equal to the cargo acceleration due to exogenous disturbances; by assumption, the values of the variables x_c and $\dot{x}_c$ at the initial time instant are known.

For further considerations, we suppose that $l^2 >> r_c^2$, which holds in most practical cases. Then the parameter a_c depends only on the wind and $a_c^{\dot{v}} \approx 0$. Hence, equation (3) can be written as

$$\ddot{x}_c \approx a_c^{\varphi}\varphi + a_c \approx -g\varphi + a_c; \quad (4)$$

here the parameter a_c represents the acceleration generated by the wind force alone.

According to this equality, under the above assumptions, the linear movement of the cargo is mainly affected only by the angular position of the suspension and the wind load. We will use equation (4) as a controlled object. Despite its substantially simplified form valid only under the above assumptions, this equation will be employed to design a control law (an approximation of the linear movement of the cargo), similar to the approach described in [11], albeit with some modification.

The natural frequency ω_0 of the angular oscillations of the cargo was found in [11] based on the parameters in Fig. 1 and equality (4). It satisfies the relations

$$\omega_0 \approx \sqrt{-a_c^{\varphi}/l} = \sqrt{\mu\, g/l} \approx \sqrt{g/l}\,;\ \omega_0 \geq \sqrt{g/l_{\max}}. \quad (5)$$

Given the crane passport data $l_{\min}$, $l_{\max}$, $v_{\max}$, the coordinates of the cargo's target point $x_{c\,\text{giv}}$, h_{giv}, and the initial positions of the trolley and cargo on the horizontal axis (they are considered to be known), we pose the following problem: under the current parametric uncertainty of the crane, cargo, and exogenous disturbances (with unknown onset times and intensities), it is required to design a control law generating a given speed of the crane trolley to be executed by the servo drive so that the cargo's horizontal movement will satisfy the conditions

$$\dot{\varphi} \to 0; \quad x_c \to \tilde{x}_{c\,\text{giv}}(l). \quad (6)$$

If $\varphi_{\text{st}} \neq 0$, the second condition in (6) requires positioning of the cargo not over the target point but so that, by the end of the control process, the cargo will be delivered to the target point by lowering without additional movements of the trolley (cargo movement along the A–TP line). In addition, conditions (6) must be fulfilled with motion dynamics close to the required one considering the speed characteristics of the servo drive. Modern asynchronous servo drives are fast enough: they have signal processing delays at a level of hundredths and tenths of a second. Therefore, we suppose that $\dot{x}_T \cong v_{\text{giv}}$ [12].

Let the main data about the suspension and cargo be provided by the combined sensor (an accelerometer and an ARS) placed on the suspension cable (similarly to the patent [3]) close to the crane trolley. See point *S* in Fig. 1 and a more detailed diagram in Fig. 2.

This placement of the combined sensor on the cable is advantageous over its location on the hook with remote information transmission proposed in [11]:

– It is possible to organize wired information transmission from the sensor and wired power supply for the sensor, a much more reliable and simple solution with reduced operating costs.

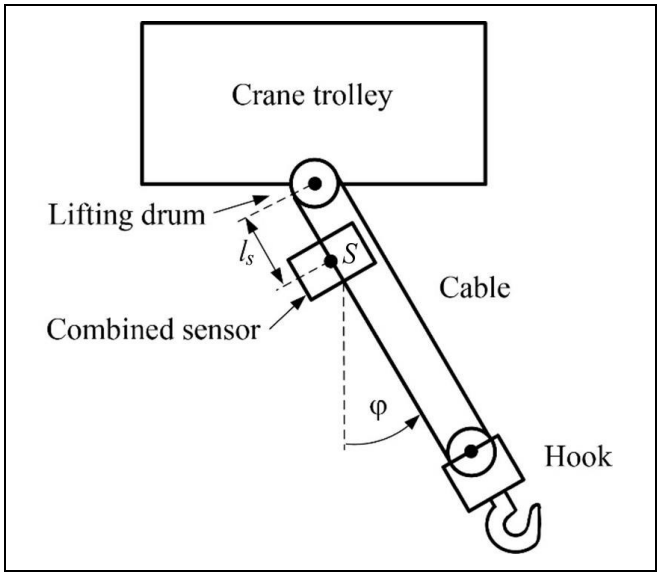

Fig. 2. The combined sensor placement on the suspension cable.

– The sensor is located in a safe place without exposure to external mechanical impacts.

– It is easy to fix the sensor sensitivity axes. (The suspension hook often rotates around the vertical axis.)

– The data obtained from this sensor are universal regardless of the kinematic scheme of the lifting mechanism [14].

– The cargo's additional secondary high-frequency motions (including the unmodeled ones) associated with the double-pendulum motions of the hook and cargo are transferred to the sensor to a lesser extent.

2. CONTROL ALGORITHM

To achieve conditions (6) with a given speed of the crane trolley, we apply the approach [11] based on the control scheme with a current parametric identification algorithm, an implicit reference model, and "simplified" adaptability conditions to track the cargo movement directly. Within this approach, the implicit reference model is an oscillating link with the values of the variables at the initial time instant t_0 equal to those of the original object:

$$\ddot{x}_m = -2\omega_m\xi_m\dot{x}_m - \omega_m^2\left(x_m - \tilde{x}_{c\,\mathrm{giv}}(l)\right),$$
$$x_m(t_0) = x_c(t_0), \quad \dot{x}_m(t_0) = \dot{x}_c(t_0), \tag{7}$$

where x_m is the variable describing the reference dynamics of the cargo motion (corresponding to the variable x_c); ω_m and ξ_m are the reference values of the natural frequency and relative damping coefficient, respectively, representing assigned parameters of the reference model [15].

First, let the values of the parameters a_c^{φ}, a_c and the variables φ, φ_{st} be known. We equate the right-hand sides of equation (4) and the first equality in (7), replacing x_m and $\dot{x}_m$ by x_c and $\dot{x}_T$, respectively, to find the control law generating the given speed v_{giv} of the crane trolley:

$$\dot{x}_T \approx v_{\mathrm{giv}} = T_x^{-1}\left[\left(\tilde{x}_{c\,\mathrm{giv}}(l) - x_c\right) - \omega_m^{-2}\left(a_c^{\varphi}\varphi + a_c\right)\right]$$
$$= T_x^{-1}\left(\tilde{x}_{c\,\mathrm{giv}}(l) - x_c\right) - 0.25T_x\xi_m^{-2}\left(a_c^{\varphi}\varphi + a_c\right), \tag{8}$$

where $T_x = 2\xi_m/\omega_m$ is the time constant of the linear movement of the trolley and cargo (the control time constant), set by the reference model.

Indeed, based on equation (4), $\ddot{x}_c$ can be substituted in formula (8) instead of the term $(a_c^{\varphi}\varphi + a_c)$. Then, in view of the notations introduced in (7), we arrive at the equality describing the cargo's behavior in the closed-loop control system with the control law (8):

$$\ddot{x}_c \approx -2\omega_m\xi_m\dot{x}_T - \omega_m^2\left(x_c - \tilde{x}_{c\,\mathrm{giv}}(l)\right). \tag{9}$$

It matches the assigned reference (7), except that the variable $\dot{x}_c$ is replaced by $\dot{x}_T$ to eliminate internal instability in the closed-loop control system.

According to Fig. 1, the tracking error of the cargo position can be represented as

$$\tilde{x}_{c\,\mathrm{giv}}(l) - x_c = x_{T\,\mathrm{giv}} - x_T, \tag{10}$$

where $x_{T\,\mathrm{giv}} = \tilde{x}_{c\,\mathrm{giv}}(0) = x_{c\,\mathrm{giv}} - \left(l_{\max} - h_{\mathrm{giv}}\right)\varphi_{\mathrm{st}}$.

Therefore, we write equality (8) in the form of the dynamics equation of the closed-loop control system:

$$T_x\dot{x}_T + x_T \approx x_{T\,\mathrm{giv}} - \omega_m^{-2}\left(a_c^{\varphi}\varphi + a_c\right). \tag{11}$$

In the absence of angular motion $(\varphi \equiv 0)$ and disturbances $(a_c \equiv 0)$, it follows that $x_T \to x_{T\,\mathrm{giv}}$ and hence $x_c \to \tilde{x}_{c\,\mathrm{giv}}(l)$ by the aperiodic law with the time constant T_x (which explains the name of this parameter).

Due to formulas (4) and (11), if a steady state $\dot{x}_T \to 0, \dot{\varphi} \to 0, \dot{x}_c \to 0, \ddot{x}_c \to 0$ is reached in the closed-loop control system, it can only be the case when $\varphi \to \varphi_{\mathrm{st}} = -a_c/a_c^{\varphi}$ and $x_T \to x_{T\,\mathrm{giv}}$. As a result, by the expression (10), $x_c \to \tilde{x}_{c\,\mathrm{giv}}(l)$.

For the closed-loop control system (2), (4), (8) with $\omega_m < \omega_0$, we have $\dot{x}_T \to 0, \dot{\varphi} \to 0, \dot{x}_c \to 0, \ddot{x}_c \to 0$ as $t \to +\infty$; this fact was established in [11]. Hence, the control objective (6) is achieved.

In crane operations, the obvious desired movement of the cargo to a given point is the process with the minimum possible control time and overshoot. As is known, for the oscillating link, this requirement corresponds to the values of the relative damping coefficient not less than 0.71 [15]. Also, the parameters ω_m and T_x in (8) should be chosen depending on the maximum speed of the drive and the required movement of the cargo. Due to these provisions, the control law (8), and equality (11) without the last term (only the aperiodic process and its maximum speed are considered [15]), we write the requirements for these parameters in the following form:

$$\begin{gathered} 0.71 \le \xi_m < 1, \quad \omega_m = \frac{2\xi_m v_{\max}}{k_{\text{tr}} \Delta x_c} < \omega_0; \\ T_x \triangleq T_x\left(\Delta x_c\right) = k_{\text{tr}} \frac{\Delta x_c}{v_{\max}} > \frac{2\xi_m}{\omega_0} = \frac{\xi_m T_0}{\pi} \overset{\xi_m = 0.8}{\approx} \frac{T_0}{4}, \end{gathered} \tag{12}$$

where k_{tr} is the positive coefficient of change of the transient time, chosen considering inequalities (12); $\Delta x_c = \left| x_{c\,\text{giv}} - x_{c_0} \right|$ is the cargo transfer distance, where x_{c_0} is the cargo's initial position on the Ox axis; $T_x\left(\Delta x_c\right)$ is the dependence of the parameter T_x on Δx_c (further denoted by T_x under a fixed value of the argument); T_0 is the period of natural pendulum oscillations of the cargo suspension.

Let us explain the choice of the coefficient k_{tr}. If $k_{\text{tr}} \ge 1$, we have $|v| \le v_{\max}$ in the transients; the higher the value of this coefficient is, the greater the control time will be. Under $k_{\text{tr}} < 1$, the transient time will decrease but the trolley speed can reach the value $v_{\max}$; at such time instants, there is no oscillation-damping control, which (of course) worsens the quality of control.

In view of the relations (5), the third inequality in (12) will hold if

$$\begin{gathered} T_x\left(\Delta x_c\right) = k_{\text{tr}} \frac{\Delta x_c}{v_{\max}} \\ > \frac{2\xi_m}{\sqrt{g}} \sqrt{l_{\max}} \overset{\xi_m = 0.8}{\approx} \sqrt{l_{\max}}/2, \end{gathered} \tag{13}$$

where the units of measurement correspond to the SI system.

Figure 3 shows the admissible domain of the parameter T_x (13) depending on $l_{\max}$ in the case $\xi_m = 1$ (the upper unshaded area). The boundary in this figure represents the minimum value of T_x.

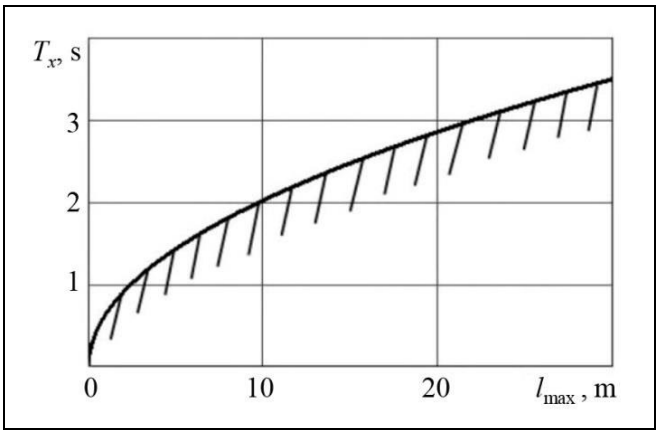

Fig. 3. The admissible domain of the parameter T_x of the control law (8) depending on the maximum length of the crane suspension under the unit relative damping coefficient of the reference model (the upper unshaded area).

Hence, the expression (13) can be used to determine the value of the parameter T_x for the control law (8) from the passport data $l_{\max}$, $v_{\max}$ of the crane installation and the cargo transfer distance Δx_c, which are known in advance. Also, this parameter can be refined based on the period of the natural oscillations of the suspension using inequality (12).

The relations (12) and (13) reflect the stability bounds of the parameter T_x in the control law (8). Obviously, the further its value lies from the bounds (i.e., the greater the value of T_x is), the more stable the closed-loop control system will be. However, the transients become longer.

Consider an example of choosing the value of the parameter T_x based on these relations. Let $l_{\max} = 10$ m, $v_{\max} = 1$ m/s, $\Delta x_c = 8$ m, $\xi_m = 0.8$, and $k_{\text{tr}} = 1$. Then, by formula (13), we obtain $T_x > 1.6$ s and $T_x = 8$ s. Due to equation (11), this solution ensures the control time $t_{\text{con}} \approx 3T_x = 24$ s to transfer the cargo to the given point [15]. (By assumption, the pendulum oscillations will be damped by the end of the transients.) When the cargo transfer distance is reduced, the values of these parameters decrease: $T_x \ge 1.6$ s, $t_{\text{con}} \ge 4.8$ s. These performance indicators agree with the time standards [16], e.g., for medium overhead cranes with a lifting capacity of up to 50 tons. Such cranes will be considered in the model example below.

The control law (8) generating the given speed is based on the exact value of the expression ($a_c^{\varphi}\varphi + a_c$), i.e., the parameters a_c^{φ}, a_c of the controlled object (4). With various types of transferred cargos, suspension

lengths, and exogenous disturbances, the values of these parameters are often unknown in practice. Another problem is the exact determination of the angle φ. The variable $\ddot{x}_c$ can be used in the control law (8) instead of $\left(a_c^{\varphi}\varphi + a_c\right)$. However, this acceleration cannot be directly measured using an accelerometer placed in the cargo's center of gravity: the accelerometer measures the sum of the apparent acceleration (generated by the support reaction due to gravity) and the desired acceleration. Application of the dependence $\ddot{x}_c \approx \ddot{x}_T + l\ddot{\varphi}$ from system (2) requires knowledge of the suspension length l and, moreover, results in a highly noisy signal due to the accelerations.

To simplify the crane control system, we propose replacing the cargo's linear acceleration in the control law (8) with the linear acceleration $\ddot{x}_s$ of the combined sensor placed at point *S* at the distance l_s from the trolley $\left(0 < l_s < l_{\min}\right)$; see Figs. 1 and 2. Similar to [11], we will use not the signal of this acceleration but its approximation based on the current parametric identification.

According to the above assumptions and formulas (2) and (4), the linear acceleration of the cargo is $\ddot{x}_c \approx \ddot{x}_T + l\ddot{\varphi} \approx a_c^{\varphi}\varphi + a_c \approx -g\varphi + a_c$. Obviously, the same acceleration for the combined sensor is equal to $\ddot{x}_S \approx \ddot{x}_T + l_s\ddot{\varphi}$. By comparing these dependencies, we easily find

$$\begin{aligned}\ddot{x}_S &\approx \ddot{x}_T + l_s\ddot{\varphi} \approx \ddot{x}_c - \left(l - l_s\right)\ddot{\varphi} \\ &\approx -g\varphi + a_c - \left(l - l_s\right)\ddot{\varphi}.\end{aligned} \tag{14}$$

In view of this equality and formula (10), we write the control law (8) by replacing the variable x_c with x_S:

$$\begin{aligned}\dot{x}_T &\approx v_{\text{giv}} = T_x^{-1}\left[\left(x_{S\,\text{giv}} - x_S\right) - \omega_m^{-2}\,\ddot{x}_S\right] \\ &\approx T_x^{-1}\left(x_{c\,\text{giv}} - \left(l_{\max} - h_{\text{giv}}\right)\varphi_{\text{st}} - x_T\right) \\ &\quad -0.25T_x\xi_m^{-2}\left(-g\varphi + a_c - \left(l - l_s\right)\ddot{\varphi}\right),\end{aligned} \tag{15}$$

where $x_{S\,\text{giv}} = \tilde{x}_{c\,\text{giv}}(l_s) = x_{c\,\text{giv}} - \left(l_{\max} - h_{\text{giv}} - l_s\right)\varphi_{\text{st}}$.

The first part of (15) directly implies the equation describing the horizontal movement of the combined sensor; it is analogous to equality (9), where the variable x_S should be substituted for x_c. Making the same substitution in the proof from [11] under the conditions of achieving the control objective will give the same conclusions for the linear movement of the combined sensor. Recall that this sensor is placed between the attachment point of the suspension on the trolley and the cargo's center of gravity on their line. Hence, based on formula (10), the control law (15) is similar to the expression (8).

To implement the control law (15) generating the given speed, in particular, it is necessary to determine the current values of the variable φ_{st} and the approximated value of $\ddot{x}_S$. Consider some possible ways to do it.

The value $\varphi_{\text{st}} \neq 0$ can be caused only by the wind effect. If an accelerometer is placed at point *S* to measure the sum of the acceleration of the support reaction due to gravity and $\ddot{x}_S$, its value will be given by $\ddot{x}_S^{\text{acc}} \approx g\varphi + \ddot{x}_S$ due to formula (14). According to the stated above, we have $\ddot{x}_S \cong 0$ for the steady-state process in the closed-loop control system. Therefore,

$$\hat{\varphi}_{\text{st}} \approx \left(\ddot{x}_S^{\text{acc}}/g\right)_{\text{lp}}, \tag{16}$$

where the upper cap "$\cap$" indicates an estimate; $\ddot{x}_S^{\text{acc}}$ is the readings of the accelerometer placed at point *S* whose sensitivity axis is orthogonal to the suspension cable; the subscript "lp" denotes low-pass filtering to eliminate the transient components.

To approximate $\ddot{x}_S$, we will use the integral of the signal from an ARS placed at point *S*. Due to the known drift of the sensor, this procedure will generate the dependence

$$\int_t \omega_{\text{ARS}}dt \approx \varphi + \Delta\omega_{\text{dr}}t + \varphi_0, \tag{17}$$

where ω_{ARS} is the data from the ARS with the sensitivity axis parallel to the suspension rotation axis; $\Delta\omega_{\text{dr}} \approx \text{const}$ is the drift of this sensor; φ_0 is the initial value of the angle φ.

We build an approximating current parametric identification algorithm based on equations (14) and (17), generating the equality

$$\begin{aligned}z \triangleq \ddot{x}_S + g\int_t \omega_{\text{ARS}}dt &= \ddot{x}_T + l_s\ddot{\varphi} + g\int_t \omega_{\text{ARS}}dt \\ &\approx \hat{a}_1 t + \hat{a}_2 = \hat{\boldsymbol{\theta}}^{\text{T}}\mathbf{y},\end{aligned} \tag{18}$$

where z is the response of the identification object; $\hat{a}_1$ and $\hat{a}_2$ are the estimates of the parameters $a_1 = g\Delta\omega_{\text{dr}}$ and $a_2 = g\varphi_0 + a_c$, respectively; $\hat{\boldsymbol{\theta}}^{\text{T}} = \left[\hat{a}_1,\ \hat{a}_2\right]$ is the vector of the estimated parameters; $\mathbf{y}^{\text{T}} = \left[t,\ 1\right]$ is the vector of the factor variables; finally, T denotes transpose.

For the signal $\ddot{x}_s$, the approximating estimate $\hat{\ddot{x}}_s$ is given by

$$\hat{\ddot{x}}_s = -g\int_t \omega_{\text{ARS}}dt + \hat{a}_1 t + \hat{a}_2 . \qquad (19)$$

Note that the expression (18) neglects the term $(l_s - l)\ddot{\varphi}$ of the original equality (14). This is dictated by experiments with the closed-loop control system with the control law (20) generating the given speed under different parameter values: due to the rapid convergence $\ddot{\varphi} \to 0$ and the self-tuning properties of the control system, the estimate of this term has little effect on the resulting motion dynamics. Moreover, with this solution, the estimate $\hat{\ddot{x}}_s$ is ahead of $\ddot{x}_s$, making the closed-loop control system with natural delays more stable.

We apply the recurrent least-squares method with the forgetting factor [17] as a current identification algorithm:

$$\begin{cases} \hat{\boldsymbol{\theta}}_i = \hat{\boldsymbol{\theta}}_{i-1} + \mathbf{P}_i \mathbf{y}_i \varepsilon_i, \ \varepsilon_i = z_i - \hat{\boldsymbol{\theta}}_{i-1}^{\mathrm{T}} \mathbf{y}_i \\ \mathbf{P}_i = \left[\mathbf{P}_{i-1} - \mathbf{P}_{i-1}\mathbf{y}_i \mathbf{y}_i^{\mathrm{T}} \mathbf{P}_{i-1} \left(\beta + \mathbf{y}_i^{\mathrm{T}} \mathbf{P}_{i-1} \mathbf{y}_i \right)^{-1} \right] \Big/ \beta \\ \mathbf{P}_0 = \vartheta \mathbf{E}_2, \ \beta < 1, \ \beta \to 1, \end{cases} \qquad (20)$$

where $i = 1, 2, 3, \ldots$ denotes the i th time instant with a step Δt; ε is the identification residual; $\mathbf{P}_i$ is the covariance matrix of the parameter estimation errors, of dimensions 2 × 2; β is the assigned forgetting factor of previous measurements to track the time-varying desired parameters; ϑ is a large positive number determining the initial rate of variation of the parameter estimates; finally, $\mathbf{E}_2$ is an identity matrix of dimensions 2 × 2.

Note that the identification algorithm (20) uses linearly independent factor variables on any time interval. This makes the resulting estimates stable, generally ensuring the stability of the identification algorithm as well [18].

Therefore, we replace the control law (15) generating the given speed with another one based on the current estimates $\hat{\varphi}_{\text{st}}$, $\hat{a}_1$, $\hat{a}_2$, and $\hat{\ddot{x}}_s$ using the expressions (16), (18)–(20):

$$\begin{aligned} \dot{x}_T \approx v_{\text{giv}} = T_x^{-1} \left(x_{c\ \text{giv}} - \left(l_{\max} - h_{\text{giv}} \right) \hat{\varphi}_{\text{st}} - x_T \right) \\ -0.25 T_x \xi_m^{-2} \left(-g\int_t \omega_{\text{ARS}}dt + \hat{a}_1 t + \hat{a}_2 \right). \end{aligned} \qquad (21)$$

Following recommendations [18], a low-pass filter should be applied to the law (21) to eliminate high-frequency components in the closed-loop system while maintaining control accuracy. Such motions may arise due to the double pendulum suspension formed by the hook and cargo in some cases [10].

For the linearly independent elements of the vector function $\mathbf{y}_i$ on a sliding time interval, the identification residual of the algorithm (20) very quickly—in a few iterations—converges to the neighborhood of zero under a sufficiently small step Δt and an appropriately chosen value of the parameter β; moreover, it will remain therein, despite that the parameter estimates may be far from the true values [18]. Thus, the algorithm (20) ensures the condition

$$\hat{\ddot{x}}_s \to \ddot{x}_s , \qquad (22)$$

i.e., approximates the variable $\ddot{x}_s$ even if the current parameter estimates are inaccurate.

In other words, the estimates can be substituted into the law (21) from the very beginning of the identification algorithm. In addition, the control objective is achieved for the current parameter estimates and $\hat{\ddot{x}}_s$, as was proved in [11].

Considering this algorithmic support of the adaptive control system of the crane trolley, we propose the following information sensors and additional algorithmic operations:

– an encoder on the crane trolley or $x_T \approx \int_t v_{\text{giv}}dt + x_{T_0}$ due to the servo drive properties mentioned above, where the last term is the initial position of the trolley;

– the combined sensor (an accelerometer and an ARS) placed on the suspension cable close to the crane trolley at point *S* at the distance l_s (Fig. 3), by analogy with the patent [3]; the accelerometer is used to obtain the signal $\hat{\varphi}_{\text{st}}$ by formula (16), whereas the ARS provides the signal ω_{ARS};

– filtering on the real differentiating link of the signal $(v_{\text{giv}} + l_s \omega_{\text{ARS}})$ to form the signal $\ddot{x}_s$, used in equality (18); it gives an approximate values of the desired signal without the ARS drift effect [15].

Note that these formulas correspond to the relative smallness of the cargo's radius of inertia, . Computer simulations show that otherwise, e.g., in the case , the closed-loop control system also provides high-quality control. In particular, the reason is the good approximation properties of the identification algorithm (20), which ensures condition (22).

3. A MODEL EXAMPLE

To compare the new approach with the previously known solution [11], we modeled the proposed control system under the same conditions as in the cited paper, even with a somewhat expanded variation range of the crane and cargo parameters; for this purpose, the dependencies (1), (18)–(21) and the expressions (5)–(7), (12), (13), and (16) were used. In particular, numerical simulation was carried out in Matlab/Simulink/SimMechanics, and the differential equations were solved by the Runge–Kutta method of the fourth and fifth orders with a step of 0.01 s.

Consider control of the trolley of a typical medium crane with the following parameters: $m_T = 450$ kg, $m_c = 100 - 50\,000$ kg, $l_{\min} = 3$ m, $l_{\max} = 15$ m, $l_s = 1$ m, $r_c = 0.2 - 5$ m, $x_{c\,\mathrm{giv}} = 10$ m, and $h_{\mathrm{giv}} = 0$. The friction force is viscous: $f_{\mathrm{fri}} = k_{\mathrm{fri}} v$, where $k_{\mathrm{fri}} = 0.3\,\mathrm{N\cdot s/m}$. The servo drive generating the speed $\dot{x}_T$ of the crane trolley according to a given value v_{giv} is described by an aperiodic link with the unit gain and a time constant of 0.1 s. It has additional nonlinearities: a time delay of 0.03 s and the output signal constraints $v_{\max} = 0.67$ m/s and $|\dot{v}| \le 3\ \mathrm{m/s^2}$. Many of these parameters match the standard [19] and the variety of typical cargo. The speed control law (21) is filtered on an aperiodic link with the unit gain and a time constant of 1 s.

Assume that a step wind disturbance with an intensity of 5% of the cargo weight affects the cargo at the time instant 50 s, corresponding to $\varphi_{\mathrm{st}} = 2.9^\circ$. (This disturbance is smoothed by an aperiodic link with a time constant of 1 s.)

The angular velocity and linear acceleration were measured using an MPU-6050 micromechanical sensor. The data contain the centered Gaussian noise with RMS errors of 0.1 deg/s (angular velocity) and 0.1 $\mathrm{m/s^2}$ (acceleration), $\Delta\omega_{\mathrm{dr}} = 0.03\ \mathrm{s^{-1}}$ [20]. The accelerometer readings had a constant bias (the accelerometer's "zero") of 0.17 $\mathrm{m/s^2}$, which corresponds to an accelerometer's angular setup inaccuracy of about 1°. The linear movement x_{T} of the trolley was determined by an encoder with similar noise with an RMS error of 0.01 m.

The identification algorithm (20) has the following parameter values: $\Delta t = 0.01$ s (the same for the law (21)), $\vartheta = 10$, and $\beta = 0.985$. In the identification algorithm, the variable $\ddot{x}_s$ is replaced by its approximate description $\ddot{x}_s(s) \approx \dfrac{s}{0.5s+1}\left(v_{\mathrm{giv}}(s) + l_s \omega_{\mathrm{ARS}}(s)\right)$, where s denotes the Laplace transform variable.

In the operating modes presented, the natural frequency of the crane (the relations (5)) varies in the range $\omega_0 = 0.8 - 1.8\ \mathrm{s^{-1}}$ or $T_0 = 3.49 - 7.85$ s. Therefore, in view of the expressions (12) and (13), the parameter values of the law (21) are taken as $\xi_m = 0.9$, $k_{\mathrm{tr}} = 2/3$, $\omega_m = 0.2\ \mathrm{s^{-1}}$, and $T_x = 10$ s. The dependence (16) has the form $\tilde{\varphi}_{\mathrm{st}}(s) \approx \dfrac{1}{2s+1}\left(\ddot{x}_s^{\mathrm{acc}}(s)/g\right)$.

We compared the behavior of the closed-loop control system on the variables x_T and x_c with the variable x_m, representing the output of the model corresponding to the reference motion (7) with the above parameters: $\ddot{x}_m = -2\omega_m \xi_m \dot{x}_m - \omega_m^2 \left(x_m - \tilde{x}_{c\,\mathrm{giv}}(l)\right)$.

Figure 4 shows the simulation results under the average values of the crane operating mode parameters with the minimum suspension length $l = 3$ m: $m_c = 5000$ kg and $r_c = 2$ m. For the other values of the last two parameters from their ranges (see above), the curves turn out to be almost the same, deviating merely by units of percent.

Next, Fig. 5 presents the simulation results under the average values of the crane operating mode parameters with the maximum suspension length $l = 15$ m: $m_c = 5000$ kg and $r_c = 2$ m. Just as in the previous case, the other values of the last two parameters from their ranges (see above) lead to almost the same curves. The constant positioning error of the cargo at the target point is about 0.28 m. If the angular setup of the accelerometer in the combined sensor has no error, this positioning error will disappear.

For other suspension lengths, the results are of intermediate character. They confirm the theoretical considerations: with a large variety of cargo parameters, its movement is close to the behavior of the assigned reference with reaching the target point at the loading height with a small error proportional to the angular setup inaccuracy of the accelerometer in the combined sensor (the presence of the uncompensated "zero" of the accelerometer). When a step wind disturbance occurs, it is parried. Note that these properties are obtained under the current parametric uncertainty. The linear movement of the cargo is close to the reference. The transient time for a 10 m load transfer is about 25 s. The angular deviation of the cargo suspension does not exceed 2°. Such properties were obtained for other values of the crane parameters without changing the control algorithm.

According to the simulation results, compared to the approach described in [11], the control system design method proposed above has similar properties in terms of the quality of cargo transfer control, the damping of pendulum oscillations, and disturbance parrying.

Also, the crane control algorithms under consideration have been validated on an experimental setup to investigate cargo calming on overhead cranes with asynchronous servo drives [21], demonstrating similar properties.

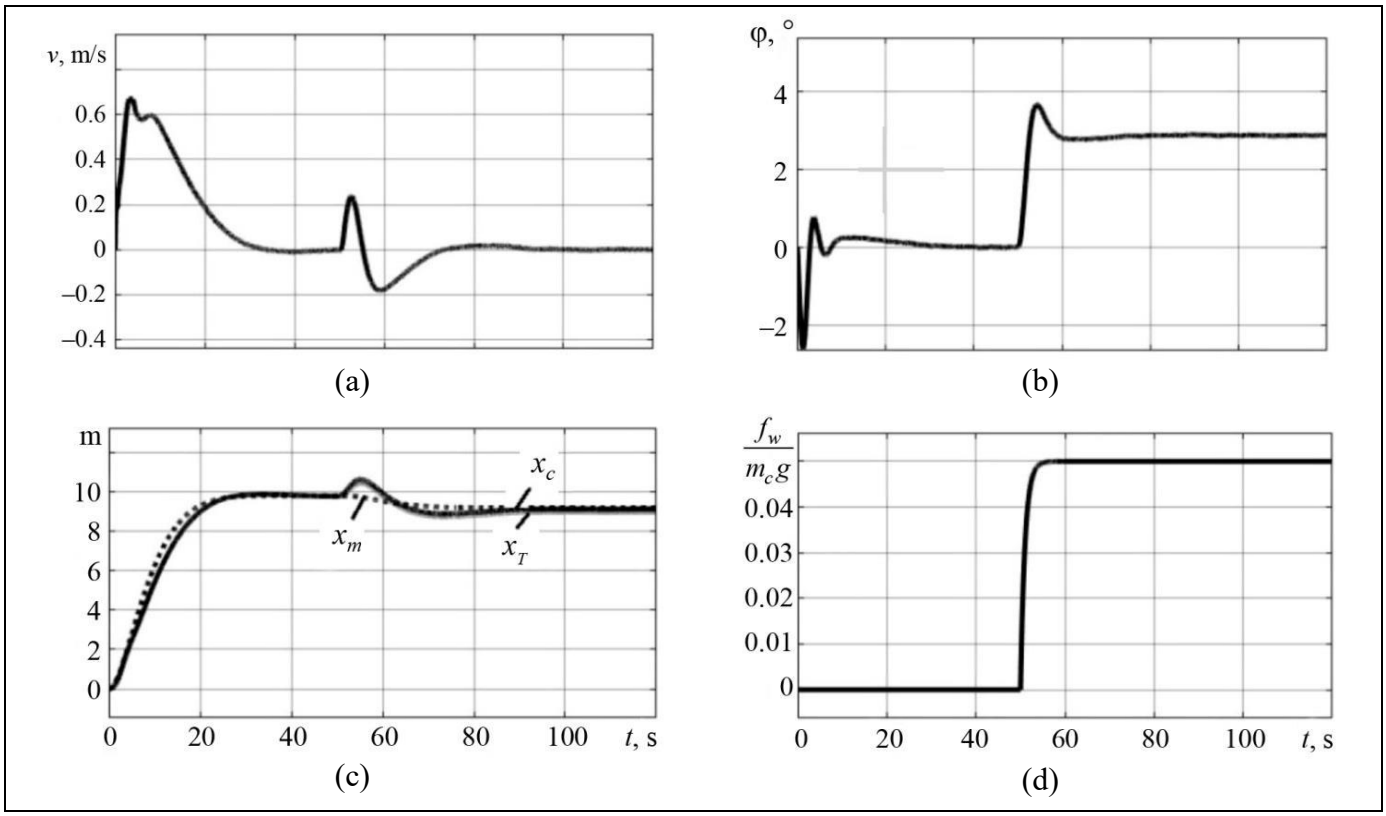

Fig. 4. Simulation results for the adaptive control system of a medium overhead crane with the minimum suspension length (3 m): (a) trolley speed, (b) the deviation angle of the suspension, (c) the linear movements of the trolley, cargo, and reference output, and (d) the relative wind force.

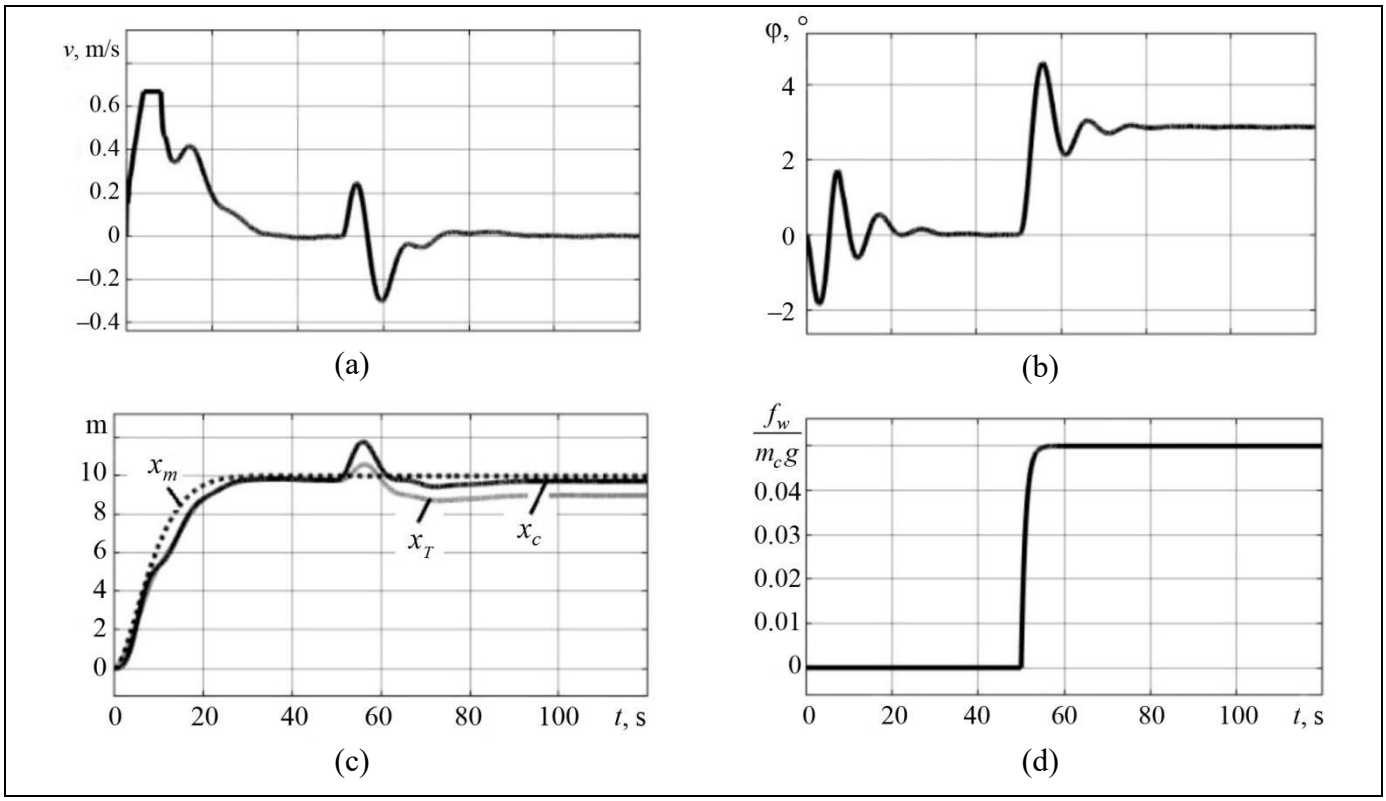

Fig. 5. Simulation results for the adaptive control system of a medium overhead crane with the maximum suspension length (15 m): (a) trolley speed, (b) the deviation angle of the suspension, (c) the linear movements of the trolley, cargo, and reference output, and (d) the relative wind force.

CONCLUSIONS

The studies have shown the effectiveness of the presented identification-based control solution for an overhead crane and its implementability using modern equipment. The approach proposed in this paper yields a control law with similar properties as in [11], but it provides additional advantages:

– the safer, more reliable, and less costly (in terms of maintenance operations) placement of information sensors;

– application of an effective current parametric identification procedure based on the least-squares method with the guaranteed stability of this algorithm;

– possibility to select control law parameters based on the passport data of a crane installation;

– no requirement to pre-tune the control system before each start (to determine the ARS drift). Therefore, inexpensive mid-class sensors can be used in the control system.

If the crane control system is intended for automated crane control, it has to be tuned at the mounting stage as follows:

– determining the "zero" of the accelerometer for further subtraction from the current readings to eliminate the constant cargo positioning error;

– selecting the value of the parameter and determining the dependence by the expressions (12) and (13) using the crane passport data for the control law (21) generating the given speed;

– selecting and validating the parameter values of the current identification algorithm.

– setting the parameter values of the low-pass filters.

(Note that periodic fine-tuning during scheduled maintenance work is also possible.)

If the crane control system is intended for fully automatic operation, an accelerometer with the highly stable "zero" should be used, or accelerometer readings should be periodically corrected through special procedures.

Acknowledgments. *This work was supported by the Russian Science Foundation, project no. 23-29-00654; https://rscf.ru/project/23-29-00654/.*

REFERENCES

1. Jaafar, H.I., Mohamed, Z., MohdSubha, N.A., et al., Efficient Control of a Nonlinear Double-Pendulum Overhead Crane with Sensorless Payload Motion Using an Improved PSO-Tuned PID Controller, *Journal of Vibration and Control*, 2018, no. 25(4), pp. 907–921.
2. Fadlalla, A.A.M. and Hassan, M., Dynamic Modeling and Feedback Linearization Control of a 3-D Overhead Gantry Crane System, *Proceedings of 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, Toronto, ON, Canada, 2021, pp. 1–6.
3. Savarezi, S.M., Vinati, F., Vinati, S.B., et al., Patent RU 2676210 S1, *Byull. Izobret.*, 2018, no. 36. (In Russian.)
4. Antipov, A.S. and Krasnova, S.A., Stabilization System of Convey-Crane Position via Sigmoidal Function, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2019, vol. 20, no. 10, pp. 609–614. (In Russian.)
5. Wu, X., Xu, K., Lei, M., and He, X., Disturbance-Compensation-Based Continuous Sliding Mode Control for Overhead Cranes with Disturbances, *IEEE Transactions on Automation Science and Engineering*, 2020, vol. 17, no. 4, pp. 2182–2189.
6. Drag, L., Model of an Artificial Neural Network for Optimization of Payload Positioning in Sea Waves, *Ocean Engineering*, 2016, vol. 115, pp. 123–134.
7. Qiang, H.Y., Sun, Y.G., Lyu, J.C., and Dong, D.S., Anti-Sway and Positioning Adaptive Control of a Double-Pendulum Effect Crane System with Neural Network Compensation, *Frontiers in Robotics and AI*, 2021, vol. 8, art. no. 639734.
8. Guo, W., Liu, D., Yi, J., and Zhao, D., Passivity-Based-Control for Double-Pendulum-Type Overhead Cranes, *Proceedings of IEEE Region 10 Annual International Conference*, Chiang Mai, Thailand, 2004, pp. 546−549.
9. Sun, N., Fang, Y., Wu, Y., and Chen, H., Adaptive Positioning and Swing Suppression Control of Underactuated Cranes Exhibiting Double-Pendulum Dynamics: Theory and Experimentation, *Proceedings of 31st Youth Academic Annual Conference of Chinese Association of Automation*, Wuhan, China, 2016, pp. 87–92.
10. Kruglov, S.P., Kovyrshin, S.V., and Aksamentov, D.N., Adaptive Control of Two-Pendulum Suspension of Overhead Crane, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2022, vol. 23, no. 9, pp. 451–461. (In Russian.)
11. Kruglov, S.P., and Kovyrshin, S.V., Identification-Based Speed Control of an Overhead Crane with a Reduced Cargo Transfer Model, *Control Sciences*, 2023, no. 4, pp. 25–33.
12. German-Galkin, S.G., *Komp'yuternoe modelirovanie poluprovodnikovykh sistem v MATLAB 6.0* (Computer Simulation of Semiconductor Systems in MATLAB 6.0), KORONA-Print, 2010. (In Russian.)
13. Chernous'ko, F.L., Anan'evskii, I.M., and Reshmin, S.A., *Metody upravleniya nelineinymi mekhanicheskimi sistemami* (Control Methods for Nonlinear Mechanical Systems), Moscow: Fizmatlit, 2006. (In Russian.)
14. Zhegul'skii, V.P., and Lukashuk, O.A., *Proektirovanie, konstruirovanie i raschet mekhanizmov mostovykh kranov* (Design, Construction, and Calculation of Overhead Crane Mechanisms), Kozhushko, G.G., Ed., Yekaterinburg: Ural University, 2016. (In Russian.)
15. Pervozvanskii, A.A., *Kurs teorii avtomaticheskogo upravleniya* (Course of Automatic Control Theory), St. Petersburg: Lan', 2015. (In Russian.)
16. *Time Standards for Loading and Unloading Jobs on Rail, Water, and Automobile Transport*. URL: https://legalacts.ru/doc/normativy-vremeni-na-pogruzochno-razgruzochnye-raboty-vypolnjaemye-na/?ysclid=lupk0hbixq562099001. (Accessed April 7, 2024.) (In Russian.)
17. Ljung, L., *System Identification: Theory for the User*, Prentice Hall, 1987.
18. Kruglov, S.P., *Adaptivnaya avtomatizatsiya pilotirovaniya samoletom na bol'shikh uglakh ataki na osnove uproshchennykh uslovii adaptiruemosti* (Adaptive Automation of Aircraft Piloting at Large Angles of Attack Based on Simplified Adapt-

ability Conditions), Irkutsk: Irkutsk Branch of Moscow State Technical University of Civil Aviation, 2012. (In Russian.)
19. *GOST* (State Standard) *3332-54: General-Purpose Electric Overhead Cranes with Load Capacity from 5 to 50 Tons of Medium and Heavy Operating Modes*, 1974. (In Russian.)
20. MPU-6000 and MPU-6050 Product Specification Revision 3.4, Sunnyvale, CA: InvenSense, 2013. URL: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf. (Accessed April 5, 2024.)
21. Kovyrshin, S.V., Kruglov, S.P., Butorin, D.V., and Kodenev, K.F., Experimental Installation for Development and Research of Algorithms for Stilling Load on Bridge-Type Cranes with a Control System Based on Industrial Elements, *Young Science of Siberia*, 2023, no. 3(21). URL: https://ojs.irgups.ru/index.php/mns/article/view/1503. (Accessed April 7, 2024.) (In Russian.)

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

Received April 12, 2024,
and revised June 28, 2024.
Accepted July 18, 2024.

Author information

Kruglov, Sergey Petrovich. Dr. Sci. (Eng.), Irkutsk State Transport University, Irkutsk, Russia
✉ kruglov_s_p@mail.ru
ORCID iD: https://orcid.org/0000-0001-9241-3352

Kovyrshin, Sergey Vladimirovich. Cand. Sci. (Eng.), Irkutsk State Transport University, Irkutsk, Russia
✉ sergkow@mail.ru
ORCID iD: https://orcid.org/0000-0001-5564-0951

Butorin, Denis Vital'evich. Cand. Sci. (Eng.), Irkutsk State Transport University, Irkutsk, Russia
✉ den_butorin@mail.ru
ORCIDiD: https://orcid.org/0000-0002-1160-5756

Cite this paper

Kruglov, S.P., Kovyrshin, S.V., and Butorin, D.V., An Identification-Based Control Method for an Overhead Crane with a New Combined Sensor Placement. *Control Sciences* 4, 52–62 (2024). http://doi.org/10.25728/cs.2024.4.5

Original Russian Text © Kruglov, S.P., Kovyrshin, S.V., Butorin, D.V., 2024, published in *Problemy Upravleniya*, 2024, no. 4, pp. 61–73.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences,
Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com