TECHNICAL CONDITION MONITORING METHODS TO MANAGE THE REDUNDANCY OF SYSTEMS. PART I: Built-in Control and Partition into Classes

V. N. Bukov*, A. M. Bronnikov**, A. S. Popov***, and V. A. Shurman****

*Airborne Aeronautical Systems, Moscow, Russia
**Bauman Moscow State Technical University, Moscow, Russia
***Zhukovsky–Gagarin Air Force Academy, Voronezh, Russia
****Institute of Aircraft Equipment, Zhukovsky, Russia

*✉ v_bukov@mail.ru, **✉ bronnikov_a_m@mail.ru, ***✉ saga30@yandex.ru, ****✉ shurman@niiao.ru

Abstract. Redundancy management of a technical system involves a monitoring procedure (control of the current state of its components) to reconfigure the system and improve the performance and autonomy of its application. This paper initiates a four-part survey of the state-of-the-art monitoring methods for redundancy management. Part I is mainly devoted to the analysis of voting schemes, fidelity rules, control codes, and program control, representing the most widespread monitoring methods in modern technical systems and built-in control. In addition, we examine long-known, albeit less common, monitoring methods: diagnosis with partition into classes and diagnosis based on algebraic invariants.

Keywords: technical condition monitoring, redundancy management, diagnosis, built-in control, control codes, partition into classes, algebraic invariants.

INTRODUCTION

The increased capabilities of information and mathematical support of control processes in complex dynamic systems and equipment complexes enable a fundamentally new approach to meeting the ever-tightening requirements for their fault tolerance, particularly based on manageable redundancy [1, 2]. One task of redundancy management in advanced systems is to perform a monitoring procedure, i.e., track current changes [3] in the operational readiness of the components of such systems [4, 5] in order to reconfigure them whenever necessary.

Technical condition monitoring consists in observing the state of a given object during an interval of its life cycle (e.g., an aircraft flight). This process is based on a certain hierarchy of methods for determining the technical condition of an object, i.e., technical diagnosis [6]. According to [7], monitoring is an integral part of maintenance.

Technical diagnosis is usually a discrete sequence of technical diagnoses (diagnostic results bound to certain time instants) [6].

Diagnosing the technical condition of technical systems is a very complex problem requiring a wide range of algorithmic solutions. From an engineering point of view, the content of control (diagnosis) of systems is to detect (find) faults by available features.

Concerning the prospects of this R&D direction, the strongest results in the field of monitoring should be expected from a system approach[1], primarily based on the triunity of the following key directions: 1) reliability and operability, 2) rationality (limitation of resources used, essential, e.g., for self-checking circuits [8]), and 3) the reasonable depth (“granularity”) of the

[1] This idea was suggested by one of the paper’s reviewers, and we express sincere gratitude for it. However, the scientific substantiation and revelation of the corresponding considerations is the subject of a separate publication.

system design, optimizing the balance of large- and small-fragment partitions into system components.

The figure shows the classification of currently used and being developed diagnosis methods for dynamic systems, with keywords characterizing them rather than their conventional names. On the one hand, test and functional control have much in common in the methods and procedures used and, on the other, they possess some peculiarities not discussed in the survey.

Below we summarize the main approaches to diagnosing the faults of components of aircraft on-board equipment complexes (OECs). This survey does not claim to be exhaustive. We endeavor to treat the subject systematically and illustrate the capabilities and typical limitations of common approaches as well as their development trends. The presentation evolves from the simplest (obvious) methods to more and more complex ones, requiring the developer's knowledge of special mathematical apparatus.

1. MONITORING BASED ON BUILT-IN CONTROL

Built-in control (BiC) [9, 10] is among the most widespread modern solutions for ensuring the required fault tolerance of various technical systems. In systems of increased danger (particularly aircraft OECs), built-in control is implemented for all components (systems, subsystems, assemblies, modules, and even microcircuits).

BiC is a set of hardware or software components, introduced into systems, their parts, or functional assemblies (FAs). As a rule, they do not participate in the work of functional modules (FMs) of the system or its FAs on purpose but collect and summarize various data that objectively reflect the operability of these modules in the developer's opinion. Looking forward, BiC can be based on various monitoring methods and their combinations, covered in all four parts of the survey. Here we consider only the most widespread approaches, which are widely implemented in modern BiC.

There are two different organizational approaches to the operation of BiC: test control and functional control.

1.1. Test Control

In test control [11, 12], assemblies, devices, and the entire system are checked using special equipment, namely, generators of test (input) impacts and analyzers of output responses. Due to the need for additional

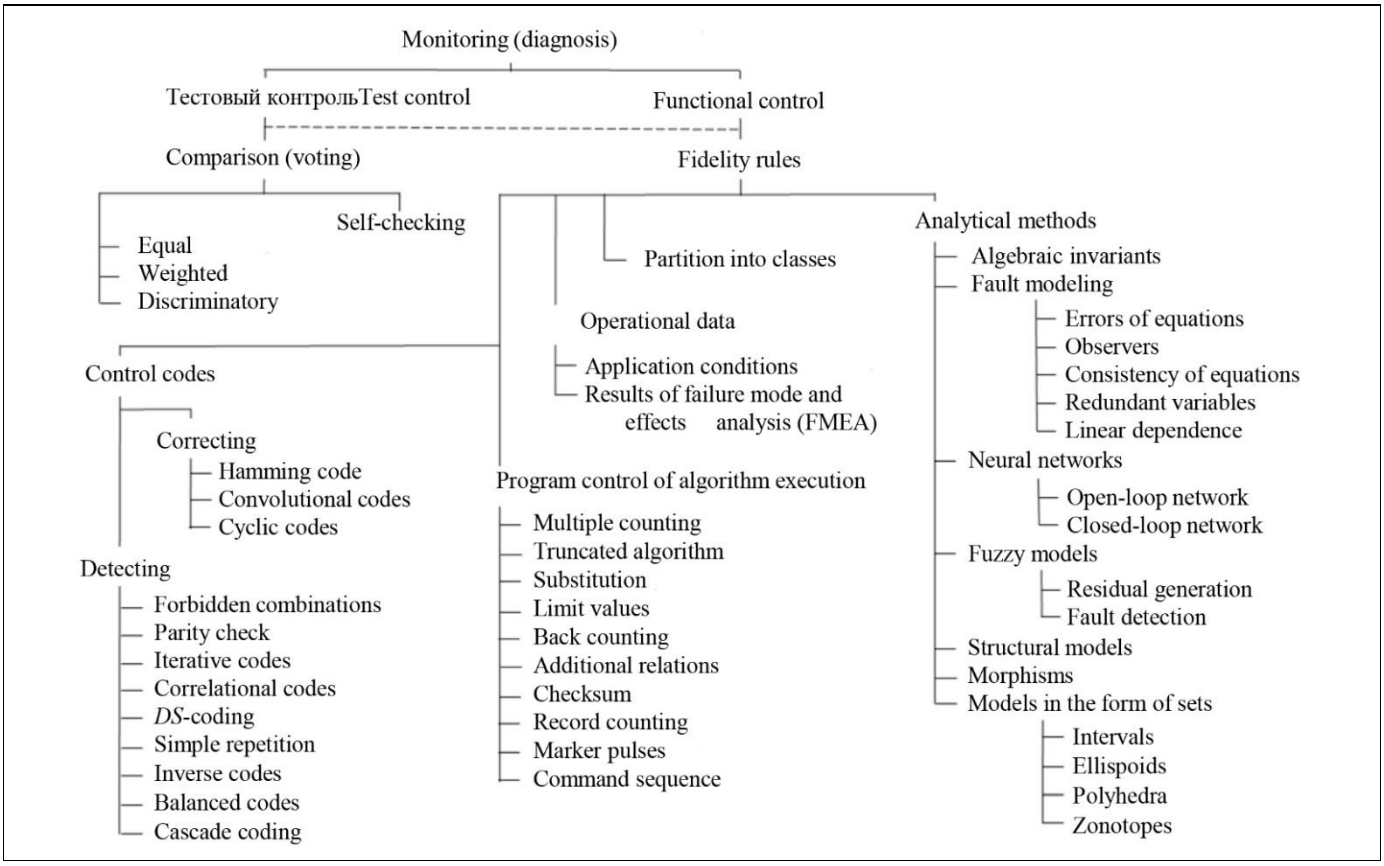

Fig. Diagnosis methods for dynamic systems: a classification.

equipment and the complexity of combining with normal operation, test methods are applied only when the controlled object is not used for its intended purpose.

In onboard conditions, testing is performed using specialized BiC tests in a background mode during special intervals (slots) allocated by the real-time operating system. The content of BiC tests is the comparison of the results of addressing (writing and reading) software-accessible resources of the computing unit, including specially organized control channels.

Testing with simulation of standard impacts is performed by a special generator of tests, and the output responses are compared with the reference ones using an analyzer.

Probabilistic testing involves a test generator of pseudorandom impacts and statistical processing of the output data; the results are compared with the reference ones obtained beforehand.

Testing with switch counting includes generating a sequence of test sets of signals at the circuit's input and calculating the number of switches at its output; the result is compared with the reference.

In *signature testing,* the controlled object is stimulated using a generator of pseudorandom impacts, and the output responses are compressed through a signature analyzer; the resulting signatures are compared with the reference signatures.

1.2. Functional Control

Functional control is performed during the intended-purpose operation of the controlled object and is generally implemented based on two main principles: voting schemes and fidelity rules.

The principal peculiarity of *comparison (voting) schemes* is the simultaneous use of several technical devices (subsystems, assemblies, or modules) identical in purpose and implementation. The diagnosis system is reduced to the means of comparing the data of these systems and selecting a preference by a given rule of comparing their outputs. Here, a common solution is the so-called quorum elements (QEs), which identify faulty modules by processing the voting results of several connected FMs. The operability of an FM is judged by a significant deviation of its output from those of same-type modules (the largest deviation or that exceeding a given threshold) [13–15].

The main peculiarities of the quorum-based method include:

– the assumption that the technical state of an FM remains unchanged within a cycle;

– the assumption that a QE is operable (never fails)[2];

– applicability to three or more FMs (in the case of two FMs, a pair of FMs becomes the controlled object, not each FM separately);

– the assumption that within the voting rules (equal, weighted, with discriminations, etc.), the operable FMs within each cycle dominate the faulty ones and the latter can be disconnected;

– a common data flow for all FMs.

A peculiar form of comparison is widely implemented in the so-called self-checking systems [8]: a set of same-type modules subjected to identical input actions is divided into pairs, and the outputs within each pair are compared with each other. A pair with matching outputs is considered to be operable; otherwise, both modules of the pair are considered to be inoperable.

The principal peculiarity of systems using *fidelity rules* (FRs) is the presence of a single diagnosed device. Depending on particular conditions and solutions, such rules can be as follows: comparing with reference (electronic) models, detecting violations of given time and (or) parametric intervals (control by parameter tolerance [13]), checking logical and other relations, calculating different-order invariants, etc.

The main peculiarities of the method of FRs include the following:

– Within each cycle, the operability of an FM does not change.

– By assumption, an element implementing FRs is operable. (If there is a reference model, it is operable.)

– This method is applicable to any number of FMs.

– By assumption, the input and output data contain sufficient information.

– Each FM has a separate data flow.

The recent direction [16] stands somewhat apart. It can be called FM monitoring based on operational data with recording of application conditions. By assumption, a special element (chip) is structurally and functionally connected directly to an FM to gather and accumulate data on the conditions of its use and storage. Such a chip stores different parameters (FM data) and sends them to the monitoring module, in particular:

– passport information,

– test results at different stages of the life cycle,

[2] The matter concerns a conceptual solution; however, multilevel majorization schemes are known that shift this constraint to higher levels of comparison of the results.

– statistics of operation indicators and characteristics (estimates of the achieved accuracy, remaining life, energy indicators, etc.),

– statistics of external impacts during intended use, storage, and routine maintenance.

The monitoring module is responsible for analyzing the incoming data and judging FM operability based on the analysis results.

1.3. Advantages and Conditions of Using Built-in Control

Thus, we summarize the common peculiarities (limitations) of BiC with different degrees of occurrence:

– weak[3] assumptions about the unchanged operability of the controlled devices within the monitoring cycle;

– strong[4] assumptions about the operability of control systems or their major devices;

– the requirement for a minimum admissible or large number of FMs (in the case of quorum or majority control);

– the requirement that operable FMs dominate inoperable FMs;

– the fast disconnection of faulty FMs;

– the sufficient informativeness requirement for all processes in FMs.[5]

The main advantage of using BiC (in the current form) to monitor the components of a redundant OEC is the well-established technologies of their creation and application in practice.

2. USE OF CONTROL CODES

A specific direction of built-in control of digital devices is the use of control codes to detect and correct errors in digital data [11, 17–23].

Block codes are most widespread: a symbolic sequence at the source output is divided into blocks (codewords, or code combinations) containing the same number of symbols. Any code can detect and correct errors (is noise-resistant) if some of its codewords are not used for information transmission [24]. In other words, a noise-resistant code must be redundant. Nevertheless, two types of noise-resistant codes are distinguished: codes with error detection and codes with error correction (correcting codes).

Error detection consists in identifying the transformation of the received or read (allowed) codeword into the so-called forbidden one. Note that the errors related to its transformation into another authorized codeword are not detected.

Error correction is a more complex operation: all forbidden codewords are divided into disjoint subsets, and each subset is assigned to one of the allowed codewords. Thus, the belonging of the resulting forbidden codeword to a subset is interpreted as the corresponding allowed codeword. If the resulting forbidden codeword belongs to neither of the subsets, the error will be detected but not corrected.

The error detection and correction properties of codes are characterized by the detection (K_{det}) and correction (K_{cor}) coefficients, $K_{\text{det}} > K_{\text{cor}}$, which have a probabilistic nature.

Detecting codes include, e.g., the following common codes.

A forbidden combination check code detects combinations of bit values in a codeword that are declared invalid (e.g., accessing a non-existent address).

A parity check code can be treated as a special case of a forbidden combination check code. It is formed by adding one non-informative bit to the information bits storing a codeword (mod2 convolution, supplementing the number of units in a code to oddness, checked at each exchange between registers). A parity check code is simple in technical realization and detects errors of odd multiplicity.

Iterative codes belong to the class of *product codes* and can be written as rectangular matrices or tables (can be built from matrices of higher dimensions). The information symbols recorded by rows and columns can be encoded by a noise-resistant code of the same or different types.

In a particular case, iterative codes are an evolution of parity check codes: they are used for the separate parity checking of rows, columns, and other structures of stored and transmitted data arrays. Such codes are characterized by simplicity and efficiency in detecting multiple errors. As an illustration, we present the control principle of a two-dimensional matrix array with parity checking by rows, columns, and the principal diagonal [22]:

$$
\begin{array}{ccccc|c}
1 & 0 & 1 & 1 & 1 & \mathrm{par}_{1j}=0 \\
0 & 0 & 1 & 0 & 1 & \mathrm{par}_{2j}=0 \\
1 & 1 & 1 & 0 & 0 & \mathrm{par}_{3j}=1 \\
0 & 1 & 0 & 0 & 1 & \mathrm{par}_{4j}=0 \\
1 & 1 & 1 & 0 & 1 & \mathrm{par}_{5j}=0 \\
\hline
\mathrm{par}_{i1}=1 & \mathrm{par}_{i2}=1 & \mathrm{par}_{i3}=0 & \mathrm{par}_{i4}=1 & \mathrm{par}_{i5}=0 & \mathrm{par}_{ii}=1
\end{array} .
$$

[3] This assumption is not crucial in practice.

[4] This assumption significantly narrows the applicability of the approach.

[5] The need for this requirement will be illustrated in part III of the survey.

Correlational codes involve an additional control bit introduced for each information bit of a word so that the entry "01" corresponds to the initial information value "0" and the entry "10" to the value "1." In this case, the codes "00" and "11" are signs of data distortion.

DS-coding [23] is a kind of using correlation codes. It is implemented via two parallel channels of sequential code transmission. If there is a bit value change from 0 to 1 or from 1 to 0 in the main channel *D*, no bit value change will occur in the control channel *S*. And vice versa, if there is no bit value change in the channel *D*, the bit value will change from 0 to 1 or from 1 to 0 in the channel *S*. The receiver controls the absence of either a simultaneous change or constancy of bit values in both channels. If this condition is violated, a data transmission error will be detected. This type of coding is characterized by a minimum error detection delay (one stroke).

A simple repetition code involves the repeated transmission of codewords. If they coincide, then the absence of an error is confirmed; otherwise, errors are detected.

An inverse code is a modification of a simple repetition code: in the case of an odd number of units in a source word, its inversion is added to it. The inverse code is received in two stages. In the first stage, the units in the base word are summed. If the number of units is even, the control bits will be received without change; if odd, the control bits will be inverted. In the second stage, the control and information bits are summed modulo 2. The zero sum indicates the absence of errors. If the sum is non-zero, the received word will be rejected.

Balanced codes are the simplest block codes in which allowed words contain a fixed number of units. They are used mainly for data transmission via communication channels.

Cascade coding. The advantages of different coding methods can be combined by applying cascade coding. In this case, information is first encoded with one code and then with another, resulting in a *product code*.

Correcting codes include, but are not limited to, the following.

A Hamming code is one of the most widely known classes of linear codes [17, 20, 24, 25]. In this code, the bits with numbers representing the degree of two are control bits, and the rest are information bits. As a rule, the maximum possible number of information bits is determined based on the number of control bits. There are no Hamming codes with one control bit; a Hamming code with two control bits contains one information bit, etc. The result is achieved by repeatedly checking the received combination for parity. Each check must cover part of the information bits and one of the redundant bits. When transmitting data (writing data to memory), the values of the control bits are formed and written; when receiving (reading) these data, the control bits are again formed and compared with the original ones. If all the newly calculated control bits coincide with the received ones, then the message contains no errors; otherwise, an error is detected and, if possible, this error is corrected.

In *convolutional codes* [26], control bits are formed based on several information words and decoding is performed based on several codewords. The principle of convolutional codes can be compared with error correction by sense. Convolutional codes are also called lattice or trefoil codes.

In *cyclic codes* [17–19], a cyclic permutation of any codeword containing both information and control bits results in a word belonging to the same cyclic code. The formalism of operations over polynomials is used to construct cyclic codes: polynomials corresponding to the received codewords must divide by their generator without a remainder. The presence of a remainder indicates an error. If the number of errors does not exceed the calculated value, then the remainder depends on the configuration of errors and can be used for their correction. Cyclic codes allow simplifying the circuit implementation of encoding and decoding devices by using shift registers.

3. PROGRAM METHODS TO CONTROL ALGORITHM EXECUTION

The advantages of software tools include versatility, flexibility, and relatively low cost. At the same time, they require specific software packages for implementation.

The *multiple counting* method assumes that the control task is solved two or more times. In the simplest case, the coinciding results are a sign of the correct solution. A deeper approach may involve various algorithms to process the results, including voting schemes (majorized estimates). Additional memory and counting time are required.

Control by *the truncated algorithm* is intended to reduce the cost of multiple counting. It applies to the cases when the task has a simplified (reduced) algorithmic variant, and therefore a less accurate but substantially similar result can be formed. Such a variant may not satisfy the customer as the solution of the original task, but it is acceptable for assessing the correctness of the full algorithm.

The limit value check method.[6] It can be used in problems with a priori estimation of admissible ranges of the solution. Often such ranges are determined for separate checkpoints (places in the action sequence of an algorithm). This method can be treated as a variant of the truncated algorithm when it is applied to calculate the bounds of possible solutions.

The substitution method. If the algorithm to be checked solves mathematical equations, then traditionally an effective check is to substitute the resulting solution into the original equations. An admissibly small residual (the difference between the left- and right-hand sides) of the equations allows judging the correct solution. Unlike multiple counting, substitution reveals systematic (programming) errors. In addition, substitution is usually less labor-intensive than multiple counting.

The back-counting method. In some tasks, it is possible to determine the initial data by the result obtained, and the correctness of counting can be checked accordingly. The corresponding costs can sometimes be smaller than those of direct counting.

Checking by *additional relations*. In this method, it is possible to introduce relations between various parameters of the main problem, described by exact or statistical formulas. Generally speaking, it is a simplified version of the analytical methods: the method of invariants and the method of redundant variables (see part II of the survey).

The *checksum method* consists in summing up all words of an array (commands, data) and then saving the sum in a definite part of the array. In the interest of control, repeated summation and comparison with the checksum saved are performed. It is applied mainly when transferring data and uploading/downloading programs.

The record counting method is to calculate and memorize the records being executed, i.e., the datasets precisely defined. Later, when handling the data, the counting is repeated and the result is compared with the original one. This method detects losses or omissions in data processing.

Marker pulse control allows tracking the passage of certain positions by a computational process or the completion of counting. The generation of appropriate time points (markers) must be provided in the algorithm being implemented. In case of violating the prescribed marker sequence or exceeding the waiting time, the counting is interrupted and a decision on further actions is made (recalculation, use of reserve variants, or task stop).

[6] A kind of parameter tolerance control [12].

A specific case of the marker pulse method is the multiple (three to five times) sending of *a data retrieval request*. An error is fixed under no response to all the requests sent. In case of receiving a response to any request, the data transfer[7] process is considered to be fault-free.

Control of the execution sequence of commands and program modules is carried out by dividing programs into sections. Then one of the following methods is applied:

- For each section, the convolution is calculated (by counting the number of operators, by signature analysis, by using codes) and then compared with the pre-calculated value.
- Each section is assigned a certain codeword (section key), which is written to the selected RAM cell before the section execution starts and is checked at the end of this section. The nodes of branching programs are checked using keys and cyclic sections are checked by the number of cycle repetitions.

Unlike these heuristic methods, the diagnosis methods described below proceed from a relatively deeper mathematical analysis of the system diagnosed.

4. DIAGNOSIS WITH PARTITION INTO CLASSES

The mathematical formulation of the control problem was given in [27], where faults were searched in an electrical circuit. The main element of this formulation is a rectangular table of faults containing S feature rows and D state columns. Consider a fixed subset R_D of columns. If R_D is a partition of the set of state-columns into classes[8], then formally the problem is to determine a partition R_S on the set of feature rows to obtain a bijective mapping

$$R_D \Leftrightarrow R_S \,. \tag{1}$$

Condition (1) is intuitively clear: on elements from the set D as a test, one constructs at least one element from R_D and assigns to it, by the if-and-only-if implication, an element from R_S.

Various modifications of this approach have become widespread [28].

The methods of this approach involve the model of a diagnosed object described by the table of fault functions R_j^i (see the general form below). Here, D denotes the set of technical states of the object; $d_0 \in D$

[7] Data fidelity is controlled separately.

[8] By definition, classes are either disjoint or completely coincident.

is its fault-free (operable) state; $d_i \in D$, $i = \overline{1, n}$, are faulty (inoperable) states. Each inoperable state corresponds to a certain fault (failure, defect) $s_i \in S$ and conversely.

Fault functions

R		D				
		d_0	$\cdots$	d_i	$\cdots$	d_n
S	s_1	R_1^0	...	R_1^i	...	R_1^n
	$\vdots$	...	...	...	...	...
	s_j	R_j^0	...	R_j^i	...	R_j^n
	$\vdots$	...	...	...	...	...
	s_k	R_k^0	...	R_k^i	...	R_k^n

In this table, S is the set of features $s_j \in S$, $j = \overline{1, k}$; R_j^i is a fault function compactly written by the formula

$$R_j^i = \Psi^i(s_j)\,. \tag{2}$$

It represents the system behavior in an analytical, graphical, tabular, or other form. A more detailed description includes input and output signals, lists of elementary checks, and initial conditions (for dynamic objects) [28].

The general formula (2) can be further specified in the light of achieving the desired depth of diagnosis. Often the explicit formula (2) is adopted only for the fault-free state R_j^0, and the faulty states are described with respect to this state.

By assumption, all faults possess detectability and distinguishability: all faults can be unambiguously identified and separated from other faults via some set Π of elementary checks. In this case, by a simple enumeration of elementary checks $\pi_k \in \Pi$, one can partition the fault table into disjoint subsets D_υ, $\upsilon = \overline{1, \lambda}$:

$$\bigcup_{\upsilon=1}^{\lambda} D_\upsilon = D,\;\; D_\upsilon \cap D_\mu = \varnothing,\;\; \upsilon \neq \mu\,. \tag{3}$$

Fault tables are used to build both diagnostic algorithms and a physical model of the object that implements diagnostic schemes.

For the same table of fault functions and a given partition of the set D into subsets D_υ, it is generally possible to construct several complete non-redundant tests $T \subset \Pi$ (sets of elementary checks π). The content of many solutions of this approach is to minimize the number of elementary checks. Various tools are used for this purpose: dividing diagnosis tasks into direct and inverse (determining the technical state d_i via a given elementary check π_k and determining the set of checks $\{\pi_k\}$ that distinguish a given pair of faults d_m and d_n, respectively), constructing fault trees, splitting inputs and outputs, etc.

5. DIAGNOSIS BASED ON ALGEBRAIC INVARIANTS

Control with calculating algebraic invariants [28] belongs to analytical methods due to using analytical information (mathematical description) about the operation of the controlled object. It consists in checking some algebraic relations (control conditions) for the set of object's output signals, supplemented (if necessary) by one or more redundant signals. The invariance of control conditions is that, in the absence of faults, they must hold for any input signals and at any time instant.

The operation of a diagnostic device can be briefly described as follows. The device receives input U and output Y signals of the object under check. Based on these signals, auxiliary signals Z are generated to satisfy, together with the signals Y, an algebraic equation resolved with respect to the variable Δ (called the syndrome):

$$\Delta = \Phi(Y, Z) = 0. \tag{4}$$

The syndrome is invariant with respect to the vector of input signals U. If condition (4) is violated ($\Delta \neq 0$), then the occurrence of a fault is judged. In practice, due to the presence of admissible errors in measurements and calculations, the control condition (4) holds approximately, and diagnosis is performed according to the inequality $|\Delta| \leq \varepsilon$, where ε specifies the output signal tolerance for the diagnostic device. Objects possessing such algebraic invariants are called objects with natural redundancy. Examples are objects that must move along a certain (in particular, phase) trajectory (on a sphere, in a plane, etc.) in a fault-free state. The advantage of such systems is the minimum necessary a priori information about the object and additional diagnostic means. The majority of fault detection schemes have artificially created redundancy.

Now we analyze the sensitivity of the syndrome (4) to faults. Let the syndrome be an m-dimensional vector $\Delta = [\Delta_1 \;\; \cdots \;\; \Delta_m]^{\mathrm{T}}$, and let the possible faults of the object be formalized by an n-dimensional vector $F = [f_1 \quad \cdots \quad f_n]^{\mathrm{T}}$. In this case, equation (4) is written as

$$\Delta = \Phi(Y, Z, F) = 0. \quad (5)$$

Obviously, for the syndrome Δ to respond to a single fault $f_i \neq 0$, it suffices to satisfy the nonzero sensitivity condition

$$\exists j : \frac{\partial \Phi_j}{\partial f_i} \neq 0,$$

which is ensured by design solutions. In the case of a multiple fault (when several faults f_i occur simultaneously), the additional condition

$$\begin{bmatrix} \partial\Phi_1 / \partial f_1 & \cdots & \partial\Phi_1 / \partial f_n \\ \vdots & \ddots & \vdots \\ \partial\Phi_m / \partial f_1 & \cdots & \partial\Phi_m / \partial f_n \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \neq 0$$

(no mutual compensation (5) for the effects of these faults) must be valid. Therefore, under $m = n$ and the maximum rank of the Jacobi matrix, all simultaneous faults f_i are detected. If the number m of the syndrome components Δ_j is smaller than the number n of faults f_i, then the mutual compensation of these effects is possible and, consequently, they will be omitted.

CONCLUSIONS

A classification of diagnosis methods has been proposed, and the content and peculiarities of built-in control have been described. Engineering heuristic monitoring methods have been considered. Monitoring methods based on voting schemes, fidelity rules, control codes, functional control software, fault tables, and algebraic invariants have been briefly characterized. Part II of the survey will deal with diagnosis methods based on classical fault modeling of the diagnosed system. In part III, we will analyze diagnosis methods based on neural networks, fuzzy and structural models, and models in the form of sets. Finally, part IV will be devoted to new approaches to technical diagnosis and combinations of different models and methods.

REFERENCES

1. Bukov, V.N., Bronnikov, A.M., Ageev, A.M., et al., The Concept of Controlled Redundancy of On-Board Equipment Complexes, *Trudy 16-oi Vserossisikoi nauchno-prakticheskoi konferentsii "Nauchnye chteniya po aviatsii, posvyashchennye pamyati N.E. Zhukovskogo"* (Proceedings of the 16th All-Russian Scientific and Practical Conference "Scientific Readings on Aviation, Dedicated to the Memory of N.E. Zhukovsky"), Moscow, 2019, pp. 17–33. (In Russian.)
2. Bukov, V.N., Ageev, A.M., Evgenov, A.V., and Shurman, V.A., *Upravlenie izbytochnost'yu tekhnicheskikh sistem. Supervizornyi sposob upravleniya konfiguratsiyami* (Redundancy Management of Technical Systems. The Supervision Method of Configuration Management), Moscow: INFRA-M, 2023. (In Russian.)
3. *The Great Russian Encyclopedia 2004–2017*. URL: https://old.bigenc.ru/economics/text/2227291?ysclid=lt6tp1tio4782244915 (Accessed February 29, 2024; in Russian.)
4. Pouliezos, A.D. and Stavrakakis, G.S., *Real Time Fault Monitoring of Industrial Processes*, Dordrecht: Kluwer Acad. Publishers, 1994.
5. *DO-297. Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations*. Washington: RTCA Inc., 2005.
6. Naumenko, A.P., *Teoriya i metody monitoringa i diagnostiki: Materialy lektsii* (Theory and Methods of Monitoring and Diagnosis: Lectures), Omsk: Omsk State Technical University, 2017. (In Russian.)
7. *GOST* (State Standard) *R 27.605–2013: Reliability in Engineering. Equipment Maintainability. Diagnostic Check*, Moscow: Standartinform, 2014. (In Russian.)
8. 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.)
9. *GOST* (State Standard) *R 56079-2014: Aircraft Items. Flight Safety, Reliability, Testability and Maintainability Indices*, Moscow: Standartinform, 2014. (In Russian.)
10. Dolbnya, N.A., Internal Controls Board Computer System Running Real Time Operation System as an Iterative Aggregate, *Vestnik of Samara State Aerospace University*, 2012, no. 5 (36), pp. 224–228. (In Russian.)
11. Arikan, E., Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels, *IEEE Transactions on Information Theory*, 2009, vol. 55, no. 7, pp. 3051–3073.
12. Bugaeva, A.A. and Denisenko, V.V., Testing Process, Methods and Types of Software Testing, *Sinergiya Nauk*, 2022, no. 72, pp. 92–102. (In Russian.)
13. *Diagnostirovanie i prognozirovanie tekhnicheskogo sostoyaniya aviatsionnogo oborudovaniya* (Technical Condition Diagnosis and Prediction for Aircraft Equipment), Sindeev, I.M., Ed., Moscow: Transport, 1984. (In Russian.)
14. Zemlyanyy, E.S. and Tektov, M.V., The Implementation of Quorum-Element for Parametric Failure Detection, *Izvestiya Tula State University. Technical Sciences*, 2023, no. 6, pp. 104–109. (In Russian.)
15. Savina, M.G., Musonov, V.M., Khudonogov, V.P., and Seslavin, V.S., The Application of Quorum-Elements for Control of Single Type Sensors, *Reshetnev Readings*, 2013, vol. 1, pp. 377–379. (In Russian.)
16. Dzhandzhgava, G.I., Dyadischev, A.V., and Garifov, R.Sh., On a Concept for Technical Condition Monitoring of Methods Used for Analyzing Physical Media, *Ideas and Innovations*, 2018, vol. 6, no. 3, pp. 64–68. (In Russian.)
17. Kudryashov, B.D., *Osnovy teorii kodirovaniya* (Fundamentals of Coding Theory), St. Petersburg: BKhV-Peterburg, 2016. (In Russian.)
18. Telpukhov, D.V., Demeneva, A.I., Zhukova, T.D., Khrushchev, N.S. The Research and Development of Automation Systems for the Concurrent Error Detection Combinational

Circuits, *Electronic Engineering. Series 3: Microelectronics*, 2018, no. 1 (169), pp. 15–22. (In Russian.)

19. Mytsko, E.A., Osokin, A.N., and Malchukov, A.N., *CRC Checksum. A Study of CRC Checksum Calculation Algorithms*, Saarbrucken: LAP LAMBERT Academic Publishing, 2013.
20. Pavlov, A.A., Tsar'kov, A.N., Sorokin, D.E., et al., Analysis of the Effectiveness of Test-Error Correction of Random Access Memory, *Izvestiya of the Institute of Engineering Physics*, 2015, no. 4 (38), pp. 47–57. (In Russian.)
21. Selivanov, E.P. and Mitin, D.D., Option Method of Construction Noise Proof Hamming Code, *Contemporary Information Technologies*, 2006, no. 3, pp. 34–37. (In Russian.)
22. Pavlov, A.A., Tsarkov, A.N., Romanenko, A.Yu., and Mikheev, A.A., Method of Error Correction in Devices for Transmitting and Processing Information of Telecommunications Systems under the Influence of External and Internal Interference, *Aerospace Instrument-Making*, 2021, no. 8, pp. 13–24. (In Russian.)
23. *GOST* (State Standard) *R 70020-2022: Space Technology. Interfaces and Protocols of High-Speed Device Communication and Complexing of Onboard Spacecraft Systems. SpaceWire-RUS*, Moscow: Russian Institute for Standardization, 2022. (In Russian.)
24. Asokan, R. and Vijayakumar, T., Design of Extended Hamming Code Technique Encryption for Audio Signals by Double Code Error Prediction, *Information Technology and Digital World*, 2021, vol. 3, no. 3, pp. 179–192.
25. Bae, W., Han, J.W., and Yoon, K.J., In-memory Hamming Error-Correcting Code in Mersister Crossbar, *IEEE Trans. on Electron. Devices*, 2022, vol. 69, no. 7, pp. 3700–3707.
26. Zhuravlev, V.G., Kuranova, N.Yu., and Evseeva, Yu.Yu., *Pomekhoustoichivye kody* (Noise-Immune Codes), Vladimir: Vladimir State University, 2013. (In Russian.)
27. Chegis, I.A. and Yablonskii, S.V., Logical Methods of Control of Work of Electric Schemes, Trudy Mat. Inst, Steklov., 1958, vol. 51, pp. 270–360. (In Russian.)
28. *Osnovy tekhnicheskoi diagnostiki* (Fundamentals of Technical Diagnosis), vols. 1 and 2, Parkhomenko, P.P., Ed., Moscow: Energiya, 1976. (In Russian.)

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

Received November 10, 2024,
and revised March 25, 2025.
Accepted April 10, 2025.

Author information

Bukov, Valentin Nikolaevich. Dr. Sci. (Eng.), JSC Airborne Aeronautical Systems, Moscow, Moscow, Russia
✉ v_bukov@mail.ru
ORCID iD: https://orcid.org/0000-0002-5194-8251

Bronnikov, Andrei Mikhailovich. Dr. Sci. (Eng.), Bauman Moscow State Technical University, Moscow, Russia
✉ bronnikov_a_m@mail.ru
ORCID iD: https://orcid.org/0009-0009-1216-3521

Popov, Aleksandr Sergeevich. Cand. Sci. (Eng.), Zhukovsky–Gagarin Air Force Academy, Voronezh, Russia
✉ saga30@yandex.ru

Shurman, Vladimir Aleksandrovich. JSC Research Institute of Aviation Equipment, Zhukovsky, Russia
✉ shurman@niiao.ru

Cite this paper
Bukov, V.N., Bronnikov, A.M., Popov, A.S., and Shurman, V.A., Technical Condition Monitoring Methods to Manage the Redundancy of Systems. Part I: Built-in Control and Partition into Classes. *Control Sciences* **2,** 2–10 (2025).

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