









# Mathematical Method for Processing SCADA Information and Diagnostic Flows

Olena Syrotkina<sup>1</sup> , Mykhailo Aleksieiev<sup>1</sup> , Borys Moroz<sup>1</sup> , Iryna Udovyk<sup>1</sup> ,  
Andrii Martynenko<sup>1</sup> , and Viktoriia Hnatushenko<sup>2</sup> 

<sup>1</sup> Dnipro University of Technology, 19 D. Yavornytskoho Ave., Dnipro 49005, Ukraine  
syrotkina.o.i@nmu.one

<sup>2</sup> National Metallurgical Academy of Ukraine, 4 Gagarina Ave., Dnipro 49600, Ukraine

**Abstract.** This paper addresses the issue of creating and applying mathematical methods for processing and analyzing large flows of diagnostic information. It specifically refers to automating the monitoring of SCADA performance in real time as well as minimizing the time and computational resources involved in the diagnostics of its subsystems. We reviewed and analyzed modern methodologies and mathematical methods used to solve these particular problems. The mathematical method was developed to work with an ordered data structure known as “*m*-tuples based on Ordered Sets of Arbitrary Cardinality (OSAC)”. It characterizes the operability of SCADA backbone nodes. We also derived a set of functional dependencies between *m*-tuples based on their location in the ordered data structure. It allowed the mathematical model to be formed under specific conditions so as to minimize computational resources when performing a set of operations on data structure elements. This also included performing automatic SCADA self-diagnostics. Displayed within the illustration of the Boolean graph is an example of one of its operations on the Boolean elements based on the mathematical method developed. We obtained logical conclusions regarding how these mathematical methods influenced the minimization of the computational resources for SCADA subsystem diagnostics. Prerequisites were created for scientifically-based methods and algorithms for automatic self-recovery of SCADA operability in real time after reversible failures.

**Keywords:** SCADA · Fault detection · Backbone node · Big data · Reduction methods · Data organizational structure · Ordered set of arbitrary cardinality · Boolean graph · Minimization of time and computational resources

## 1 Introduction

Modern SCADA systems are widely employed in various industries as well as in the energy sector. These are distributed multi-level and multi-user complex hardware/software systems operating in real time [1]. There is an overwhelming necessity in the industry to develop methods of reliable and timely analysis to scrutinize large flows of diagnostic low-level information generated by SCADA during its operation when failures occur. The flow of diagnostic information is a complex data structure. It

includes, for example, the following attributes: the diagnostic message timestamp, the network address of the backbone node, the diagnostic message classification code, the identifier of the software process that generated this diagnostic code, etc.

Therefore, the urgent task of SCADA diagnostics is to create methods for analyzing and processing multiple streams of diagnostic information generated by SCADA automatically in the event any malfunctions or failures occur [2–4]. It is known that flows of diagnostic information in distributed SCADA systems can contain thousands of messages per second. This leads to a requirement for specialized methods of diagnostics for the system's operability in order to receive a reliable diagnosis in real time.

## 2 Analysis of Recent Publications

SCADA is a complex hardware/software system which has a multilevel, multitasking and multiuser architecture operating in real time. Modern SCADA are dynamic and difficult-to-formalize diagnostic systems with a changing structure and functionality in the process of their life cycle.

There is a negative impact on the quality control for the Technological Control Object (TCO) due to certain problems in SCADA maintenance. These are due to low-level diagnostics and the limited ability to restore SCADA operability after reversible failures.

Methods and models of failure diagnostics in automated systems are considered in the works of the authors [2–4].

Work [2] is focused on utilizing, to the maximum extent possible, large amounts of data that are systematically registered in SCADA. However, these data are not scrutinized for fault detection, diagnosis and reconfiguration. The methodology of Case Base Reasoning (CBR) is proposed to solve this issue. This work describes the following containers in which knowledge is stored in CBR: vocabulary, retrieval and adaptation knowledge. Knowledge acquisition and representation is further needed by other reasoning methods, such as expert systems. This work also provides a review of CBR terminology and basic concepts.

Work [3] discusses two complementary techniques for end-to-end stream processing: Data Stream Management Systems (DSMS) and Streaming Data Warehouses (SDW). In network monitoring, an SDW may store traffic streams that have been pre-aggregated or otherwise pre-processed by a DSMS. It summarizes the differences between Database Management Systems (DBMS) and DSMS. The authors also compared DSMS and SDW. It is important to note that DSMS needs to meet the following requirements: it must process a data stream using limited working memory and must support multi-query optimization.

Work [4] describes the development of a data acquisition system which takes advantage of open source streaming data solutions. This was developed in response to the Big Data paradigm and is especially relevant where data standardization is not normally considered. This work encompasses four areas: standardizing vocabulary, faceted browsing of data, enhanced data discovery, and machine-to-machine interoperability. The first challenge in standardizing the output of raw data streams is to provide an extensible system for harmonization of heterogeneous raw data inputs to homogeneous outputs.

### 3 Aim of Research

The aim of the research is to create a mathematical method for processing large flows of diagnostic information generated by a distributed SCADA system. This method is required to minimize the time and computational resources involved.

In order to achieve this aim we selected an ordered data structure known as “ $m$ -tuples based on OSAC”. This data structure describes, in general, a Boolean template. The Boolean template represents an ordered set of all subsets within an ordered basis set of arbitrary cardinality for any data type. The given data structure allows us to create an arbitrary algorithm for ordering the basis set data of certain data types when representing a template class. This algorithm can be adopted for various specific tasks of working with these data.

### 4 Main Terms and Definitions of Given Data Organizational Structure

The given data structure “ $m$ -tuples based on OSAC” includes the following parameters:

$X$  – ordered basis template set with cardinality  $n$ ;

$2^X$  – Boolean of set  $X$ ;

$Y_m^n$  – ordered subset of Boolean  $2^X$ . Its elements are  $m$ -tuples consisting of elements of set  $X$ ;

$y_{m,j}^n$  –  $m$ -tuple, an element of set  $Y_m^n$ ;

$n$  – cardinality of ordered basis set  $X$ ;

$m$  – tuple length;

$j$  – index (ordinal number) of  $m$ -tuple in ordered set  $Y_m^n$ ;

$k_m^n$  – cardinality of set  $Y_m^n$ .

A more detailed description of the main terms and definitions as well as mathematical methods to work with the given data structure is shown in works [5–7].

### 5 Problem Statement in General Terms

It is necessary to create a mathematical method to find the solution to system (1). This method would be applicable for any correct combination of parameter values  $n, m1, m2, j1, j2, \eta$  with minimization of the time and computational resources for data processing

and analysis.

$$\left\{ \begin{array}{l} y_{m1,j1}^n \text{ op } y_{m2,j2}^n \\ \text{op} \in \{\subset, \cup, \cap\} \\ 1 \leq m1 < m2 \leq n \\ 1 \leq j1 \leq k_{m1}^n \\ 1 \leq j2 \leq k_{m2}^n \\ 0 \leq \eta \leq j1 \\ k_{m1}^n = \binom{n}{m1} \\ k_{m2}^n = \binom{n}{m2} \end{array} \right. , \quad (1)$$

where  $m1, m2$  are tuple lengths;  $j1, j2$  are indexes (ordinal numbers) of  $m$ -tuples in ordered sets  $Y_{m1}^n$  and  $Y_{m2}^n$ ;  $\eta$  is an ordinal number of element  $y_{m1,j1}^n$  in tuple  $y_{m2,j2}^n$ .

In order to perform operations on  $m$ -tuples, it is necessary to develop a series of methods  $A^{op}(y_{m1,j1}^n, y_{m2,j2}^n)$ .

The main idea in uniting these methods is to derive a set of functional dependencies  $\{f_\gamma(n, m1, m2, j1, j2, \eta)\}$  to solve system (2).

$$\left\{ \begin{array}{l} J2_{m2,\eta}^n(y_{m1,j1}^n) \leftarrow \{f_\gamma(n, m1, m2, j1, j2, \eta)\} \\ J2_{m2}^n(y_{m1,j1}^n) = \bigcup_{\eta=1}^{j1} J2_{m2,\eta}^n(y_{m1,j1}^n) \\ (j2 \in J2_{m2}^n(y_{m1,j1}^n)) ? \text{true} : \text{false} \end{array} \right. , \quad (2)$$

where  $J2_{m2,\eta}^n$  is a set of indexes  $j2$  defining elements  $y_{m2,j2}^n$  for one of subsets  $Y_{m2}^n$  of Boolean  $2^X$  for which the condition  $0 < \eta \leq j1$  is fulfilled.  $J2_{m2,\eta}^n$  is a subset  $J2_{m2}^n$  for which  $\eta = \text{const}$ .

## 6 Main Part

### 6.1 General Method Description

In works [5–7] the authors consider mathematical methods for performing operations (see formula 1) on Boolean elements  $2^X$ . These elements are represented by  $m$ -tuples based on an ordered basis set  $X$  of arbitrary cardinality  $n$  for the initial conditions  $m1 = 1; j1 = \{1, 2, 3\}$ .

In this paper we consider a developed mathematical method  $A_{m1,j1}^{\subset}(y_{m2,j2}^n)$  to calculate the result of operation  $y_{m1,j1}^n \subset y_{m2,j2}^n$  for the given initial conditions  $m1 = 1; j1 = 4$ .

$$(y_{m1=1,j1=4}^n \subset y_{m2,j2}^n) = (A_{m1=1,j1=4}^{\subset}(y_{m2,j2}^n)) ? \text{true} : \text{false}$$

We check the correctness of input data for method  $A_{1,4}^{\subseteq}(y_{m2,j2}^n)$  in accordance with system (3).

$$\begin{cases} m1 = 1, j1 = 4 \\ m1 < m2 \leq n \\ k_{m1}^n = \binom{n}{m1} = \frac{n!}{m1! \cdot (n-m1)!} \\ k_{m2}^n = \binom{n}{m2} = \frac{n!}{m2! \cdot (n-m2)!} \\ 1 \leq j1 \leq k_{m1}^n \\ 1 \leq j2 \leq k_{m2}^n \end{cases} \quad (3)$$

We denote parameter  $p_\eta$  as a potential variant to find element  $y_{m1,j1}^n$  in the  $\eta^{th}$  place in tuple  $y_{m2,j2}^n$ . We define the constraints that affect the range of variation  $p[j1]$  according to system 4.

$$\begin{cases} p_{\eta=1} = (m2 \leq n - j1 + 1)?1 : 0 \\ p_{\eta=2} = (n - j1 \geq m2 - \eta)?1 : 0 \\ p_{\eta=3} = (\eta \leq m2 < n)?1 : 0 \\ p_{\eta=4=j1} = (m2 \geq j1)?1 : 0 \end{cases} \quad (4)$$

The task of method  $A_{1,4}^{\subseteq}(y_{m2,j2}^n)$  is to form a set  $J2_{m2}^n(y_{m1,j1}^n)$  from subsets  $J2_{m2,\eta}^n(y_{m1,j1}^n)$  taking into account constraints (see system 4) according to the following algorithm:

$$\begin{aligned} J2_{m2}^n(y_{m1,j1}^n) &= \emptyset, \\ \forall \eta \in [1; j1] &\rightarrow (J2_{m2}^n(y_{m1,j1}^n) = (p_\eta > 0) ? (J2_{m2}^n(y_{m1,j1}^n) \cup J2_{m2,\eta}^n(y_{m1,j1}^n)) : NOP) \end{aligned} \quad (5)$$

Set  $J2_{m2}^n$  contains  $k_{m2}$  elements calculated using the following formula:

$$k_{m2} = \binom{n - m1}{m2 - m1} \quad (6)$$

In turn, each of the subsets included into the model of subsets  $J2_{m2,\eta}^n$  contains  $k_{m2,\eta}$  elements.

Depending on the input data of model  $A_{m1,j1}^{\subseteq}(y_{m2,j2}^n)$ , included subsets  $J2_{m2,\eta}^n$  contain  $\mu_{m2,\eta}$  groups, where  $\mu_{m2,\eta}$  is calculated for each subset  $J2_{m2,\eta}^n$  individually and can be changed in range  $[1; j1 - 1]$ . Every  $\mu_{m2,\eta}$  group in turn contains  $k\mu_{m2,\eta}$  elements.

The conditions for determining the calculation option  $v_\eta$  of the parameters  $\mu_{m2,\eta}$  and  $k\mu_{m2,\eta}$  are summarized in system (7).

$$\begin{cases} v_{\eta=4} = (p_4)?1 : 0 \\ v_{\eta=3} = (p_3)?((m2 == \eta)?2 : ((m2 < n - 1)?3 : 4)) : 0 \\ v_{\eta=2} = (p_2)?((m2 == \eta)?5 : 6) : 0 \\ v_{\eta=1} = (p_1)?((m2 < n - j1 + 1)?8 : 9) : 0 \end{cases} \quad (7)$$

We introduce an additional parameter  $d_{m2,\eta}$  to calculate the number of elements in group  $\mu_\eta$  (see formula 8).

$$\forall \eta \in [1; j1] \rightarrow d_{m2,\eta} = ((p_\eta == 1) ? \binom{n-j1}{m2-\eta} : 0) \quad (8)$$

We calculate the number of elements in subset  $J2_{m2,\eta}^n$  in accordance with system (9).

$$\left\{ \begin{array}{l} (v_{\eta=4} == 1) ? (\mu_{m2,4} = 1, k\mu_{m2,4} = d_{m2,4}) \\ (v_{\eta=3} == 2) ? (\mu_{m2,3} = j1 - 1, k\mu_{m2,3} = 1) \\ (v_{\eta=3} == 3) ? (\mu_{m2,3} = j1 - 1, k\mu_{m2,3} = d_{m2,3}) \\ (v_{\eta=3} == 4) ? (\mu_{m2,3} = 1, k\mu_{m2,3} = j1 - 1) \\ (v_{\eta=2} == 5) ? (\mu_{m2,2} = j1 - 1, k\mu_{m2,2} = 1) \\ (v_{\eta=2} == 6) ? (\mu_{m2,2} = j1 - 1, k\mu_{m2,2} = d_{m2,2}, ((k\mu_{m2,2} > 1) ? v_2 = 7)) \\ (v_{\eta=1} == 8) ? (\mu_{m2,1} = 1, k\mu_{m2,1} = d_{m2,1}) \\ (v_{\eta=1} == 9) ? (\mu_{m2,1} = 1, k\mu_{m2,1} = 1) \end{array} \right. \quad (9)$$

We calculate the number of elements  $k_{m2,\eta}$  in subset  $J2_{m2,\eta}^n$  using formula (10).

$$(\forall \eta \in [1; j1]) \rightarrow (k_{m2,\eta} = (p_\eta > 0) ? (\mu_{m2,\eta} * k\mu_{m2,\eta}) : 0) \quad (10)$$

We define several auxiliary calculation parameters:

$$\mu_{\min} = \min_{\eta=1}^{j1}(\mu_{m2,\eta}), \quad \mu_{\max} = \max_{\eta=1}^{j1}(\mu_{m2,\eta}) \quad (11)$$

$$(\forall \mu \in [\mu_{\min}; \mu_{\max}] \rightarrow \Delta_{m2,\mu} = \binom{n-\mu+1-m1}{m2-m1}) \quad (12)$$

$$r_{m2} = \binom{n-j1}{m2} \quad (13)$$

$$h_{m2,\eta,\Delta} = \binom{n-\Delta}{m2-\eta} \quad (14)$$

We summarize the options for calculating elements of subsets  $J2_{m2,\eta}^n$  (see system 9) into Table 1.

In Table 1,  $j2_{m2,\eta,\mu,1}$  is the minimum ordinal number  $j_2$  of element  $y_{m2,j_2}^n$  in the  $\mu^{th}$  subgroup of subset  $Y_{m2,\eta}^n$  inside set  $Y_{m2}^n$  for which expression 3 of system (2) is true and element  $y_{m1,j_1}^n$  is located on the  $\eta_l^{th}$  place in tuple  $y_{m2,j_2}^n$ ;  $j2_{m2,\eta,\mu,last}$  is the maximum ordinal number  $j_2$  similarly.

We define set  $J2_{m2}^n(y_{m1,j_1}^n)$  (see formula 5) based on Table 1. Solution to expression 3 in system (2) is the final formula of the whole method  $A_{1,4}^{\subseteq}(y_{m2,j_2}^n)$ .

**Table 1.** Options for calculating elements of subsets  $J2_{m2,\eta}^n$ .

$\eta$	$V_\eta$	$J2_{m2,\eta,\mu,i}$
4	1	$j2_{m2,4,1,1} = 1, (k\mu_{m2,4} > 1)?j2_{m2,4,1,last} = k\mu_{m2,4}$
3	2	$j2_{3,3,1,1} = j1 - \eta + 1, j2_{3,3,2,1} = n - 1, j2_{3,3,3,1} = k_{m2} + 1$
	3	$j2_{m2,3,1,1} = h_{m2,3,3} + 1, j2_{m2,3,3,1} = k_{m2} + 1,$ $j2_{m2,3,\mu=2,1} = j2_{m2,3,3,1} - \Delta_{m2,\mu=2},$ $(\forall \mu \in [1; j1 - 1]) \rightarrow (j2_{m2,3,\mu,last} = j2_{m2,3,\mu,1} + k\mu_{m2,3} - 1)$
	4	$j2_{n-1,3,1,last} = k_{m2}^n, j2_{n-1,3,1,1} = j2_{n-1,3,1,last} - k\mu_{m2,3} + 1$
2	5	$j2_{2,2,1,1} = j1 - \eta + 1, j2_{2,2,2,1} = n + 1, j2_{2,2,3,1} = 2 * k_{m2}$
	6	$j2_{m2,2,1,1} = k_{m2}, j2_{m2,2,2,1} = k_{m2}^n - 1, j2_{m2,2,3,1} = k_{m2}^n$
	7	$j2_{m2,2,1,1} = h_{m2,2,2} + h_{m2,2,3} + 1$ $(\forall \mu \in ([2; j1 - 1]) \rightarrow (j2_{m2,2,\mu,1} = j2_{m2,2,\mu-1,1} + \Delta_{m2,\mu})$ $(\forall \mu \in ([1; j1 - 1]) \rightarrow (j2_{m2,2,\mu,last} = j2_{m2,2,\mu,1} + k\mu_{m2,2} - 1)$
1	8	$j2_{m2,1,1,last} = k_{m2}^n - r_{m2}, j2_{m2,1,1,1} = j2_{m2,1,1,last} - k\mu_{m2,1} + 1$
	9	$j2_{n-3,1,1,1} = k_{n-3}^n$

## 6.2 An Example of Task Solution

For Boolean  $2^X$  formed on the basis of alphabetically ordered character basis set  $X = \{a, b, c, d, e, f, g, h, i, j\}$  with cardinality  $n = 10$ , it is necessary to determine the veracity of expression  $y_{1,4}^{10} \subset y_{5,50}^{10}$  by using method  $A_{1,4}^C(y_{m2,j2}^n)$ .

We have  $n = 10, m1 = 1, j1 = 4, m2 = 5, j2 = 50$ .

We need to verify the correctness of the input data according to system (3):

$$\begin{cases} m1 < m2 \leq n, 1 < 5 \leq 10 \\ k_{m1}^n = \binom{n}{m1} = \frac{n!}{m1! \cdot (n-m1)!} = \binom{10}{1} = 10 \\ k_{m2}^n = \binom{n}{m2} = \frac{n!}{m2! \cdot (n-m2)!} = \binom{10}{5} = 252 \\ 1 \leq j1 \leq k_{m1}^n, 1 \leq 4 \leq 10 \\ 1 \leq j2 \leq k_{m2}^n, 1 \leq 50 \leq 252 \end{cases} \quad (15)$$

According to (15), the input data are correct.

We determine the solution search area by using system (4). The result is presented in system (16):

$$\begin{cases} p_{\eta=1} = (m2 \leq n - j1 + 1)?1 : 0, (5 \leq 10 - 4 + 1) \rightarrow p_1 = 1 \\ p_{\eta=2} = (n - j1 \geq m2 - \eta)?1 : 0, (10 - 4 \geq 5 - 2) \rightarrow p_2 = 1 \\ p_{\eta=3} = (\eta \leq m2 < n)?1 : 0, (3 \leq 5 < 10) \rightarrow p_3 = 1 \\ p_{\eta=4=j1} = (m2 \geq j1)?1 : 0, (5 \geq 4) \rightarrow p_4 = 1 \end{cases} \quad (16)$$

We determine cardinality  $k_{m2}$  of set  $J2_{m2}^n$  (see formula 6).

$$k_{m2} = \binom{n - m1}{m2 - m1} = \binom{10 - 1}{5 - 1} = 126 \quad (17)$$

Further options for calculating the parameters (see system 7) are determined as follows:

$$\begin{cases} v_{\eta=4} = (p_4)?1 : 0, v_4 = 1 \\ v_{\eta=3} = (p_3)?((m2 == \eta)?2 : ((m2 < n - 1)?3 : 4)) : 0, v_3 = 3 \\ v_{\eta=2} = (p_2)?((m2 == \eta)?5 : 6) : 0, v_2 = 6 \\ v_{\eta=1} = (p_1)?((m2 < n - j1 + 1)?8 : 9) : 0, v_1 = 8 \end{cases} \quad (18)$$

We define a set of additional parameters  $d_{m2,\eta}$  (see formula 8).

$$\begin{aligned} d_{5,1} &= \binom{10 - 4}{5 - 1} = 15, & d_{5,2} &= \binom{10 - 4}{5 - 2} = 20, \\ d_{5,3} &= \binom{10 - 4}{5 - 3} = 15, & d_{5,4} &= \binom{10 - 4}{5 - 4} = 6 \end{aligned} \quad (19)$$

We calculate the number of elements in subset  $J2_{m2,\eta}^n$  (see formula 9).

$$\begin{cases} (v_{\eta=4} == 1)?(\mu_{m2,4} = 1, k\mu_{m2,4} = d_{m2,4}) \\ (v_{\eta=3} == 3)?(\mu_{m2,3} = j1 - 1, k\mu_{m2,3} = d_{m2,3}) \\ (v_{\eta=2} == 6)?(\mu_{m2,2} = j1 - 1, k\mu_{m2,2} = d_{m2,2}, ((k\mu_{m2,2} > 1)?v_2 = 7)) \\ (v_{\eta=1} == 8)?(\mu_{m2,1} = 1, k\mu_{m2,1} = d_{m2,1}) \end{cases} \quad (20)$$

As a result of this calculation we have:

$$\begin{cases} \mu_{5,4} = 1, k\mu_{5,4} = 6 \\ \mu_{5,3} = 3, k\mu_{5,3} = 15 \\ \mu_{5,2} = 3, k\mu_{5,2} = 20 \\ \mu_{5,1} = 1, k\mu_{5,1} = 15 \\ v_2 = 7 \end{cases} \quad (21)$$

We calculate the number of elements  $k_{m2,\eta}$  in subset  $J2_{m2,\eta}^n$  (see formula 10).

$$k_{5,4} = \mu_{5,4} \cdot k\mu_{5,4} = 1 \cdot 6 = 6$$



$$\begin{aligned}
k_{5,3} &= \mu_{5,3} \cdot k\mu_{5,3} = 3 \cdot 15 = 45 \\
k_{5,2} &= \mu_{5,2} \cdot k\mu_{5,2} = 3 \cdot 20 = 60 \\
k_{5,1} &= \mu_{5,1} \cdot k\mu_{5,1} = 1 \cdot 15 = 15
\end{aligned} \tag{22}$$

We check the correctness of the model parameter calculations:

$$\begin{aligned}
k_{m2} &= \binom{n-m1}{m2-m1} = \sum_{\eta=1}^{j1} k_{m2,\eta} \\
\binom{10-1}{5-1} &= 15 + 60 + 45 + 6
\end{aligned} \tag{23}$$

$126 = 126$  – the calculation of model parameters is correct.

We calculate the auxiliary parameters of the model (see formulas 11–14).

$$\begin{aligned}
\mu_{\min} &= 1, \quad \mu_{\max} = 3 \\
\Delta_{5,1} &= k_5 \\
\Delta_{5,2} &= \binom{n-\mu+1-m1}{m2-m1} = \binom{8}{4} = 70 \\
\Delta_{5,3} &= \binom{7}{4} = 35 \\
r_5 &= \binom{n-j1}{m2} = \binom{10-4}{5} = 6 \\
h_{5,1} &= \binom{n-j1+1}{m2-\eta} = \binom{10-4+1}{5-1} = 35, \\
h_{5,2} &= \binom{10-4+1}{5-2} = 35, \quad h_{5,3} = \binom{10-4+1}{5-3} = 21, \\
h_{5,4} &= \binom{10-4+1}{5-4} = 7 \\
h_{5,3,3} &= \binom{n-\Delta}{m2-\eta} = \binom{10-3}{5-3} = 21, \\
h_{5,2,2} &= \binom{10-2}{5-2} = 56, \quad h_{5,2,3} = \binom{10-3}{5-2} = 35
\end{aligned} \tag{24}$$

We calculate the parameters in Table 2 using the formulas from Table 1.

As a result of the calculations in Table 2, we determine subsets  $J2_{m2,\eta}^n(y_{m1,j1}^n)$ .

$$\begin{cases}
J2_{5,4}^{10}(y_{1,4}^{10}) = \{1 \div 6\} \\
J2_{5,3}^{10}(y_{1,4}^{10}) = \{22 \div 36, 57 \div 71, 127 \div 141\} \\
J2_{5,2}^{10}(y_{1,4}^{10}) = \{92 \div 111, 162 \div 181, 197 \div 216\} \\
J2_{5,1}^{10}(y_{1,4}^{10}) = \{232 \div 246\}
\end{cases}$$

**Table 2.** Calculation of elements of subsets  $J2_{m2,\eta}^n$ .

$\eta$	$V_\eta$	$J2_{m2,\eta, \mu, i}$
4	1	$j2_{m2,4,1,1} = 1, (k\mu_{m2,4} > 1)?j2_{m2,4,1,last} = k\mu_{m2,4}$
		$j2_{5,4,1,1} = 1, j2_{5,4,1,last} = 6$
3	3	$j2_{m2,3,1,1} = h_{m2,3,3} + 1, j2_{m2,3,3,1} = k_{m2} + 1,$ $j2_{m2,3,\mu=2,1} = j2_{m2,3,3,1} - \Delta_{m2,\mu=2},$ $(\forall \mu \in [1; j1 - 1]) \rightarrow (j2_{m2,3,\mu,last} = j2_{m2,3,\mu,1} + k\mu_{m2,3} - 1)$
		$j2_{5,3,1,1} = 21 + 1 = 22, j2_{5,3,3,1} = 126 + 1 = 127,$ $j2_{5,3,2,1} = 127 - 70 = 57,$ $j2_{5,3,1,last} = 22 + 15 - 1 = 36, j2_{5,3,2,last} = 71, j2_{5,3,3,last} = 141$
2	7	$j2_{m2,2,1,1} = h_{m2,2,2} + h_{m2,2,3} + 1$ $(\forall \mu \in ([2; j1 - 1]) \rightarrow (j2_{m2,2,\mu,1} = j2_{m2,2,\mu-1,1} + \Delta_{m2,\mu})$ $(\forall \mu \in ([1; j1 - 1]) \rightarrow (j2_{m2,2,\mu,last} = j2_{m2,2,\mu,1} + k\mu_{m2,2} - 1)$
		$j2_{5,2,1,1} = 56 + 35 + 1 = 92, j2_{5,2,2,1} = 92 + 70 = 162,$ $j2_{5,2,3,1} = 162 + 35 = 197,$ $j2_{5,2,1,last} = 92 + 20 - 1 = 111, j2_{5,2,2,last} = 162 + 20 - 1 = 181,$ $j2_{5,2,3,last} = 197 + 20 - 1 = 216$
1	8	$j2_{m2,1,1,last} = k_{m2}^n - r_{m2}, j2_{m2,1,1,1} = j2_{m2,1,1,last} - k\mu_{m2,1} + 1$
		$j2_{5,1,1,last} = 252 - 6 = 246, j2_{5,1,1,1} = 246 - 15 + 1 = 232$

Then  $J2_{m2}^n(y_{m1,j1}^n)$  is defined as follows:

$$J2_5^{10}(y_{1,4}^{10}) = \{1 \div 6, 22 \div 36, 57 \div 71, 92 \div 111, 127 \div 141, 162 \div 181, \\ 197 \div 216, 232 \div 246\}$$

$$(j2 = 50) \notin J2_5^{10}(y_{1,4}^{10})$$

Result: tuple  $y_{1,4}^{10}$  is not a subset of tuple  $y_{5,50}^{10}, y_{1,4}^{10} \not\subset y_{5,50}^{10}$ .

### 6.3 Optimization of the Calculation Algorithm

When solving this example, set  $J2_{m2}^n(y_{m1,j1}^n)$  was completely defined. At the same time, we performed the maximum number of calculations as a result of which all 16 parameters from Table 1 were calculated. However, to obtain the result, there is no need to completely

define set  $J2_{m_2}^n(y_{m_1, j_1}^n)$ . In this example, it is enough to calculate 4 parameters from Table 1:  $j2_{5,3,1,1}$ ,  $j2_{5,3,1,last}$ ,  $j2_{5,3,2,1}$ ,  $j2_{5,2,1,1}$ . In this case, the execution time of the algorithm decreases accordingly. Optimization of calculations consists of choosing the correct model parameters for the calculation depending on the initial conditions.

As we can see from system (9), for element  $y_{1,4}^n$  in the maximum case, we have  $g = 8$  groups of parameters for determining index  $j2$ :

$$J2_{m_2}^n(y_{1,4}^n) = \{j2_{m_2,4,1,1}, \dots, j2_{m_2,4,1,last}, \\ j2_{m_2,3,1,1}, \dots, j2_{m_2,3,1,last}, j2_{m_2,3,2,1}, \dots, j2_{m_2,3,2,last}, j2_{m_2,3,3,1}, \dots, j2_{m_2,3,3,last}, \\ j2_{m_2,2,1,1}, \dots, j2_{m_2,2,1,last}, j2_{m_2,2,2,1}, \dots, j2_{m_2,2,2,last}, j2_{m_2,2,3,1}, \dots, j2_{m_2,2,3,last}, \\ j2_{m_2,1,1,1}, \dots, j2_{m_2,1,1,last}\};$$

Index  $j2_{max}$  of the last element of set  $Y_{m_2}^n$  is calculated as follows:

$$j2_{max} = k_{m_2}^n = \binom{n}{m_2}$$

Therefore, the sequence of indexes  $J2_{m_2}^n$  of set  $Y_{m_2}^n$  can also be divided into  $g$  intervals of length  $s = j2_{max}/g$ :

$$\{1, \dots, s, s+1, \dots, 2*s, \dots, (i-1)*s+1, \dots, i*s, \dots, (g-1)*s+1, \dots, g*s\}$$

We calculate the number of the  $i^{\text{th}}$  interval for index  $j2$  according to the condition  $(i-1)*s+1 \leq j2 \leq i*s$ . After this, we choose the initial group for the calculation and define the initial calculation parameters for our example.

$$j2_{max} = k_5^{10} = 252 \\ s = j2_{max}/g = 252/8 \approx 31$$

The sequence of indexes  $j2$  divided into  $g = 8$  groups has the following form:  
 $\{1, \dots, 31, 32, \dots, 63, 64, \dots, 94, 95, \dots, 126, 127, \dots, 157, 158, \dots, 189, 190, \dots, 220, 221, \dots, 252\}$

For  $j2 = 50$  we calculate the interval number for index  $i = 2$  ( $32 \leq 50 \leq 63$ ). Therefore, for our example, we select  $i = 2$  groups of initial parameters for the calculation  $j2_{m_2,3,1,1}, \dots, j2_{m_2,3,1,last}$ .

#### 6.4 Graphical Representation for the Solution of the Task

Graphical representation for the solution of the task using method  $A_{m_1, j_1}^C(y_{m_2, j_2}^n)$  is shown in Fig. 1. Each closed curve represents an area for solving the task.

$$J2_{m_2, \eta, \mu}^n \rightarrow A_{m_1, j_1}^C(y_{m_2, j_2}^n)$$

The number of areas for solving task  $g$  varies in the range  $[gmin; gmax]$  depending on the input data. For the initial conditions  $j1 = 1$ ,  $m1 = 4$ , parameter  $g$  is determined by using system (9).

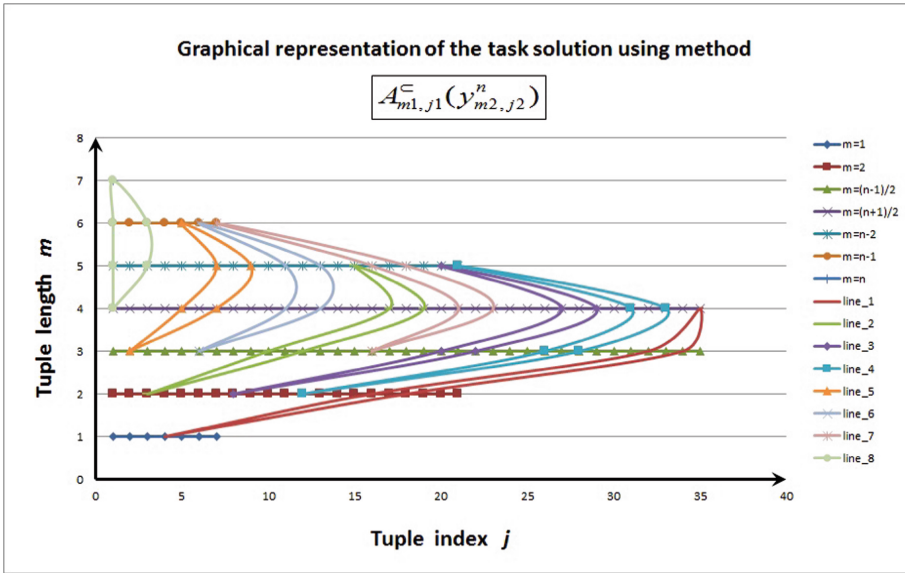


Fig. 1. Graphical representation of the task solution using method  $A_{1,4}^C(y_{m2,j2}^n)$ .

## 7 Description of the Experiment

The objective of the experimental research is to check the correctness of the detection and localization of failures in the SCADA system. This is based on the analysis of the states of the controlled parameters of the technological control object (CP TCO) when transmitting data streams through the hierarchy levels and structural modules of the SCADA system.

We denote the possible states of the controlled parameter based on the three-valued set  $\{0,1,2\}$ , where

- 0 — the value of the controlled parameter at time  $t$  is absent;
- 1 — the value of the controlled parameter at time  $t$  is unreliable;
- 2 — the value of the monitored parameter at time  $t$  is reliable.

Methods and algorithms for determining the reliability of the controlled parameter at each SCADA hierarchy level are a separate task and are beyond the scope of this article.

We determine the number of possible states of a controlled parameter when passing through SCADA hierarchy levels.

Taking the number of SCADA hierarchy levels of system  $l$  and the number of possible states of the controlled parameter at a certain hierarchy level  $k = 3$ , we obtain the number of variants of values of the vector of possible states for a controlled parameter when passing through all levels of the SCADA hierarchy as the number of locations with

repetitions of  $k$  elements in  $l$ .

$$\bar{A}_k^l = k^l$$

Experimental studies were carried out in relation to a SCADA system with  $l = 5$  hierarchy levels (primary converter – data transmission channel – remote terminal unit – data transmission channel – server).

Therefore, the full set of variants of the vector values under study contain combinations of values starting from (0,0,0,0,0) and ending with (2,2,2,2,2).

In this case, vector (0,0,0,0,0) <sub>$t_0$</sub>  is the value of the controlled parameter with the timestamp  $t_0$ , where  $t_0$  is the timestamp of the parameter measurement using the primary converter. It is known that all SCADA systems have technological timeouts  $\tau_{j,l}$  when transmitting data, signals and commands from the  $j^{\text{th}}$  structural module of the  $l^{\text{th}}$  hierarchy level to the next hierarchy level. Thus, in this case, the value of the controlled parameter is absent from the SCADA server at time  $t_1$ . Therefore,  $\Delta t = t_1 - t_0$  is the maximum timeout while waiting for the controlled parameter to be registered on the SCADA server configured in the system. Vector (2,2,2,2,2) <sub>$t_0$</sub>  represents the value of the controlled parameter which is transmitted/registered in the system at all SCADA hierarchy levels within the waiting interval configurable for each hierarchy level in the system.

To formalize the even/odd numbers of SCADA hierarchy levels in the example SCADA topology, we accept the odd SCADA hierarchy levels ( $l \% 2 = 1$ ) for registering the controlled parameter at the backbone node, the even SCADA hierarchy levels ( $l \% 2 = 0$ ) for receiving and transmitting the parameter through a data transmission channel.

The set of acceptable variants the vector values contain combines the values formed based on the following rules:

- a) the controlled parameter value at higher odd hierarchy levels should be no more than the value of this controlled parameter at lower odd hierarchy levels;
- b) the controlled parameter value for even hierarchy levels does not depend on the value of the same controlled parameter for other hierarchy levels;
- c) the controlled parameter value on odd hierarchy levels must be no more than the minimum of the values of the same controlled parameter on the previous odd and even hierarchy levels.

We form a diagnostic matrix using vectors with permissible options for the values of the controlled parameter states. For this matrix the row index corresponds to the SCADA hierarchy level and the column index corresponds to the ordinal number of the SCADA controlled parameter. The system's controlled parameters are measured by primary converters at time  $t_0$ . The diagnostic matrix is considered fully formed and ready for processing at time  $t_1$ .

The unreliability or absence of data after waiting through a timeout at a given hierarchy level is a diagnostic feature of failure detection at this hierarchy level. Configurable distribution of data streams through structural modules and hierarchy levels allows us to localize the failure of a structural module by the controlled parameter ordinal number.

It is known that distributed multi-user and multitasking SCADA systems operate with a large number of controlled parameters in real time. Therefore, in order to detect and localize failures in the system, it is necessary to process very wide diagnostic matrices where each column corresponds to one of the hierarchy levels. Diagnostic matrices are formed with a sampling rate of primary converters according to the technological processes controlled by SCADA. Their processing and analysis are carried out on the basis of the above mathematical method of working with an ordered data organizational structure “ $m$ -tuples based on ordered sets of arbitrary cardinality”.

## 8 Analysis of Results and Discussion

The mathematical method of working with the given data structure “ $m$ -tuples based on ordered sets of arbitrary cardinality” taken under consideration herein allows the space of analyzed states to be reduced for the diagnostic matrix from the total number of combinations to the permissible number of combinations  $A_f$ . The permissible state combinations are determined on the basis of the rules described above for the formation of permissible combinations of the values for the controlled parameter states.

As a result of the dependence analysis for the permissible number of combinations of the values of the controlled parameter states on the system's hierarchy levels, we derived functional dependencies to calculate  $A_f(l)$ .

We denote  $n_0(l)$ ,  $n_1(l)$ ,  $n_2(l)$  as the permissible number of value combinations of the controlled parameter states for a vector of length  $l$  where the last element of the vector determines the state of the controlled parameter on the  $l^{\text{th}}$  hierarchy level and has the value 0, 1, or 2, respectively.

We accept  $n_0(l=1) = 1$ ,  $n_1(l=1) = 1$ ,  $n_2(l=1) = 1$ .

$$n_0(l) = k \left\lfloor \frac{l-1}{2} \right\rfloor,$$

where  $\lfloor \rfloor$  - the integer part of the number.

$$\begin{aligned} n_1(l) &= n_0(l-1) + 2^{(l\%2)} \cdot n_1(l-1) \\ n_2(l) &= n_0(l-1) + n_1(l-1) + n_2(l-1) \\ A_f(l) &= \sum_{i=0}^2 n_i(l) \end{aligned}$$

The experimental studies conducted allow us to trace the dynamics of the decrease in the component part of the controlled parameter permissible states in relation to the full set of combinations depending on the number of SCADA hierarchy levels. The results of the calculations are shown in Table 3.

where

$l$  – the number of SCADA hierarchy levels in the system model;

$\overline{A}_k^l$  – the number of the controlled parameter states for the complete set of combinations;

$A_f$  – the number of the controlled parameter states for the permissible set of combinations;

**Table 3.** The number of states of the CP TCO for the complete and valid permissible set of combinations

$l$	$\bar{A}_k^l$	$A_f$	$\Delta, \%$
1	3	3	100%
2	9	6	67%
3	27	14	51.9%
4	81	25	31%
5	243	53	22%
6	729	90	12%
7	2,187	182	8.3%
8	6,561	301	4.6%
9	19,683	593	3%
10	59,049	966	1.6%
11	177,147	1,874	1%

$\Delta$  – the ratio of the permissible number of combinations of the controlled parameter states to the total number of combinations by percentage.

$$\Delta = A_f / \bar{A}_k^l \cdot 100\%$$

In this paper, the diagnostic matrix was formed to detect and localize failures in the system resulting from data flows passing through structural modules and the system's hierarchy levels. Reducing the space of analyzed states of the diagnostic matrix allows us to increase the efficiency of data processing by reducing the computing resources involved. This becomes especially significant with an increase in hierarchy levels in SCADA topology (see Table 3).

## 9 Conclusions

The method of working with the data organizational structure “ $m$ -tuples based on OSAC” developed and considered in this article was obtained by applying the rules for each element of the ascending data structure. The set of functional dependencies between these elements is derived on the basis of the uniqueness of their location in the structure.

The method developed allows us to significantly expedite the process of data processing of SCADA information and diagnostic flows, since:

- the complexity of the data structure represented by “ $m$ -tuples” does not affect data processing speed. Instead of cumbersome operations on elements of large arrays with a complex data organizational structure, the methods work with sets of integers. These are sets of indexed parameters calculated using the mathematical method for diagnosing SCADA operability as described in this paper;

- the amount of memory needed to store the data array decreases from the amount of memory needed to work with the whole Boolean compared to the amount of memory required to store the basis set. This is especially important for dynamic analysis of the diagnostic matrix in real time, as SCADA may consist of several thousand backbone nodes;
- the estimate of the execution time of the method proposed for performing operations on the Boolean elements does not depend on the cardinality of the sets representing the operands.

The practical value of the method developed lies in the possibility of processing large flows of diagnostic information in real time. The application of this method for automatic failure diagnostics is the basis for creating methods of automatic self-recovery for distributed SCADA systems after reversible failures in real time.

## References

1. Stouffer, K., Falco, J., Kent, K.: Guide to supervisory control and data acquisition (SCADA) and industrial control systems security. In: Recommendations of the National Institute of Standards and Technology: NIST Special Publication, vol. 800–82, pp. 1–170 (2013)
2. Melendez, J., Colomer, J., Rosa, J.: Expert supervision based on cases. In: 8th International Conference on Emerging Technologies and Factory Automation, pp. 431 – 440. IEEE, Antibes-Juan les Pins, France (2001)
3. Golab, L., Ozsu, M.: Data Stream Management. Morgan & Claypool (2010)
4. Leadbetter, A., Smyth, D., Fuller, R., O’Grady, E., Shepherd, A.: Where big data meets linked data: applying standard data models to environmental data streams. In: 2016 IEEE International Conference on Big Data, pp. 2929–2937. IEEE, Washington, DC, USA (2016)
5. Syrotkina, O., Alekseyev, M., Asotskyi, V., Udovyk, I.: Analysis of how the properties of structured data can influence the way these data are processed. *Naukovyi Visnyk NHU* **3**(171), 119–129 (2019)
6. Syrotkina, O., Alekseyev, M., Meshcheriakov, L., Moroz, B.: Methods of working with “big data” based on the application of “*m*-tuple” theory. *Comput. Integr. Technol. Educ. Sci. Prod.* **36**, 140–153 (2019)
7. Syrotkina, O., Alekseyev, M., Udovyk, I.: Graphical and analytical methods for processing “Big Data” based on the analysis of their properties model. *Syst. Technol.* **3**(122), 78–90 (2019)