DOI: 10.15587/978-617-8360-13-9.CH6

Svitlana Kashkevich, Oleksandr Yaryzhko,
Olena Nosyryeva, Anzhela Popova, Ihor Pimonov, Dmitry Leinyk

CHAPTER 6

CHAPTER 6

INTELLIGENT METHODS FOR EVALUATING THE STATE OF HIERARCHICAL SYSTEMS

ABSTRACT

This section of the study proposes intelligent methods for assessing the state of hierarchical systems.

During the research, the authors:

– conducted an analysis of knowledge representation models, substantiating the advantages of using production knowledge representation in expert systems. The study outlines key concepts of fuzzy expert systems and formulates a formal task for accelerating decision-making in the rule base of a fuzzy expert system;

– developed a methodology for assessment and prediction using fuzzy cognitive maps.

The novelty of the proposed methodology lies in:

– considering a corrective coefficient for the degree of uncertainty regarding the state of the object;

– adding a corrective coefficient for data noise resulting from distortions in object state information;

– reducing computational costs when evaluating object states;

– creating a multilevel and interconnected description of hierarchical objects;

– adjusting the object description due to changes in its current state using a genetic algorithm;

– enabling calculations with input data of different natures and measurement units.

The research includes the development of a visualization method for hierarchical system states. The novelty of this method lies in:

– creating a visual, multilevel, and interconnected description of the hierarchical system;

– enhancing decision-making efficiency in assessing the hierarchical system's state;

– addressing the issue of global and local extrema when evaluating the state of hierarchical systems;

– combining graphical and numerical representations of the monitored parameters of the hierarchical system's state;

– avoiding loop formation during real-time visualization of hierarchical system states.

The study also develops a method for evaluating complex hierarchical systems based on an improved particle swarm optimization (PSO). This evaluation method combines PSO with coordinate averaging and its modification by employing multiple particle swarms and integrating the Hooke-Jeeves procedure and appropriate corrective coefficients.

The novelty of this method lies in:
– creating a multilevel and interconnected description of real-time complex hierarchical systems;
– enhancing decision-making efficiency for real-time hierarchical systems assessment;
– resolving global and local extrema issues during real-time hierarchical system state evaluation;
– enabling directed searches by multiple swarm particles in a specific direction;
– considering the degree of uncertainty;
– allowing for repeated analysis of complex real-time hierarchical systems.

KEYWORDS

Artificial intelligence, heterogeneous data processing, hierarchical systems, reliability, efficiency.

Optimization is a complex process of determining a set of solutions for various functions. Many computational tasks today belong to the domain of optimization problems [1–3]. When solving optimization problems, decision variables are defined to ensure that a hierarchical system operates at its optimal point (or mode) based on a specific optimization criterion.

Optimization problems in hierarchical systems are often discontinuous, non-differentiable, and multimodal. Therefore, classical gradient-based deterministic algorithms [4–6] are not suitable for addressing the optimization tasks of organizational and technical systems.

To overcome the limitations of classical optimization algorithms in solving hierarchical system optimization problems, a significant number of stochastic optimization algorithms, known as metaheuristic algorithms, have been developed [7–11].

One type of stochastic optimization algorithm for hierarchical systems is swarm intelligence algorithms (swarm-based algorithms). These algorithms are based on the movement of a swarm and simulate its interaction with the environment to improve knowledge about the surroundings, such as discovering new food sources. The most well-known swarm algorithms include Particle Swarm Optimization (PSO), Artificial Bee Colony Optimization, Ant Colony Optimization, Wolf Pack Optimization, and Sparrow Search Algorithm [12–18].

Unfortunately, most of the aforementioned basic metaheuristic algorithms fail to balance exploration and exploitation, leading to unsatisfactory performance in solving real-world complex optimization problems [19–38].

This has motivated the development of various strategies to enhance the convergence speed and accuracy of basic metaheuristic algorithms [39–69]. One approach to improving the efficiency of decision-making with metaheuristic algorithms is their hybridization, where fundamental procedures of one algorithm are incorporated into another [70–86].

Given the above, the development of intelligent methods for assessing the state of hierarchical systems using artificial intelligence is a pressing scientific challenge. Such methods would significantly improve the efficiency of decision-making processes.

6.1 ANALYSIS OF ADVANTAGES AND DISADVANTAGES OF KNOWLEDGE REPRESENTATION METHODS IN INTELLIGENT DECISION SUPPORT SYSTEMS

This section analyzes the advantages and disadvantages of knowledge representation methods in intelligent decision support systems.

The logical model is used for knowledge representation in first-order predicate logic systems and for deriving conclusions through syllogisms.

The main advantage of using predicate logic for knowledge representation lies in its powerful inference mechanism, which possesses well-understood mathematical properties and can be directly programmed. With these programs, new knowledge can be derived from previously known facts.

Distinctive features of logical models include the unambiguity of theoretical justification and the possibility of implementing a system of formally accurate definitions and conclusions.

The main idea behind constructing logical knowledge models is as follows: all information necessary for solving applied tasks is considered as a set of facts and statements presented as formulas in a certain logic. Knowledge is represented as a set of such formulas, and obtaining new knowledge is reduced to implementing logical inference procedures.

Main advantages of logical knowledge models:

1. The foundation of logical models is the classical apparatus of mathematical logic, which methods are well-studied and formally justified.

2. Efficient inference procedures are available, including those implemented using logical programming languages.

3. Knowledge bases can store only sets of axioms, while other knowledge is derived from them using inference rules.

Frame model. The frame model, or knowledge representation model, is a systematic representation of human memory and cognition.

Frames take the form of structured components of situations, called slots. A slot may refer to another frame, thereby establishing a connection between two frames. General relationships, such as communication links, can also be established. Each frame is associated with diverse information (including procedures), such as expected procedures for a situation, ways of obtaining information about slots, default values, and inference rules.

Advantages of the frame model – representation is largely based on including assumptions and expectations. This is achieved by assigning default values to slots in standard situations. During the search for solutions, these values can be replaced with more reliable ones. Some variables are designed in such a way that the system must ask the user about their values.

Frame models ensure structural organization and connectivity. This is achieved through inheritance and nesting properties of frames, meaning that a slot may represent a system of lower-level slot names, and slots may be used as calls for executing procedures. Slot values may be refined during the processing of knowledge represented in this model. Some variables may be defined as

embedded procedures. As values are assigned to the variables, other procedures are triggered. This type of representation combines declarative and procedural knowledge.

For many subject areas, frame models are the primary method of formalizing knowledge.

Semantic networks. A semantic network is a directed graph structure where nodes represent concepts (objects, processes, situations) and edges correspond to relationships such as "is a", "belongs to", "is caused by", "is part of", "is composed of", "is like", and similar connections between pairs of concepts. Reasoning procedures used in semantic networks include network augmentation, property inheritance, and pattern matching. Distinctive features of semantic networks include class-element relationships (part-whole, class-subclass, element-set), property-value relationships (having a property, having a value), and examples of class elements (element above, element below, earlier, later).

The advantages of semantic networks include the clarity of knowledge representation, which makes it convenient to depict causal relationships between elements (subsystems) and even the structure of complex systems. The disadvantage of such networks is the complexity of reasoning and searching for subgraphs that match a specific query.

Production model. Rule-based systems are the most widely used type of expert systems today. In rule-based systems, knowledge is not represented in a declarative, static manner (as a series of true statements) but rather in the form of numerous rules that specify conclusions to be drawn or not drawn in various situations. A rule-based system consists of "IF-THEN" rules, facts, and an interpreter that manages which rule should be invoked based on the facts in the working memory.

A classical expert system embodies informal knowledge obtained from an expert.

This process of creating an expert system is called knowledge engineering and is performed by a knowledge engineer.

Rule-based systems are categorized into two main types: forward-chaining systems and backward-chaining systems. Forward-chaining systems start with known initial facts and proceed by using rules to infer new conclusions or perform specific actions. Backward-chaining systems start with a hypothesis or goal that the user aims to prove and work backward by finding rules that can confirm the truth of the hypothesis. To break down a large task into smaller, manageable fragments, new subgoals are created. Forward-chaining systems are primarily data-driven, while backward-chaining systems are goal-driven.

The working memory can contain facts about the current state of the object. A rule whose conditions are all satisfied is called activated or fired. The working list of rules may contain several activated rules. In such cases, the inference engine must choose one of the rules to execute.

The THEN part of a rule contains a list of actions to be performed once the rule is executed.

The inference engine operates in a cycle of checking and executing rules. During each cycle, multiple rules may be activated and added to the working list of rules.

Conflicts arise in the rule list when different activated rules have the same priority, requiring the inference engine to decide which rule to execute. Once all rules are executed, control is returned to the command interpreter so that the user can issue additional instructions to the expert system's command interpreter.

A notable feature of an expert system is its explanation mechanism, which allows the user to ask questions about how the system reached a specific conclusion and why certain information is needed. Rule-based systems can easily answer questions about how a conclusion was derived because the rule activation chronology and the content of the working memory can be stored in a stack.

The widespread use of rule-based systems is due to the following reasons: modular organization simplifies knowledge representation and allows for incremental development of the expert system. Explanation capabilities enable the easy creation of explanation tools through rules, as the antecedents of rules specify what is required for rule activation. The explanation tool allows tracking which rules were executed, enabling the reconstruction of the reasoning process that led to a specific conclusion.

Analogy with human cognitive processes: according to findings by Newell and Simon, rules naturally align with the way humans solve problems. When extracting knowledge from experts, it is easier to explain the structure of knowledge representation to experts because rules provide a simple and intuitive framework.

Fuzzy expert systems. The foundation of the functioning of expert systems lies in the knowledge model [9]. It contains a set of principles that describe the state and behavior of the object under study. The most widely used knowledge model of expert systems is the production model, as it is quite simple to process and understandable to the end user.

However, recently fuzzy expert systems have become widespread [11]. This type of expert system is based on a set of rules in which linguistic variables and fuzzy relationships are used to describe the state and behavior of the object under study [12]. Rules presented in this form are closest to natural language, so there is no need to use a separate knowledge engineer to create and edit the rules. They can be edited by the expert themselves almost without special training. Also, the results of such systems are presented in a limited natural language, which increases their degree of adaptation to the end user. Let's consider the organization of fuzzy expert systems in more detail.

A fuzzy expert system uses knowledge representation in the form of fuzzy productions and linguistic variables [13]. Each linguistic variable is defined using its term set, which consists of fuzzy variables [14].

Fuzzy variable. The concept of a fuzzy variable is used to describe objects and phenomena through fuzzy sets, where the membership of a specific element is defined by a membership function $\mu z(u)$, this function determines the degree to which the value u belongs to the set z [15, 17, 18]. Every fuzzy variable is characterized by a triplet $<z, U, Z>$, where z – the name of the variable, U – the universal set, Z – the fuzzy subset of U, which represents the fuzzy constraint on the value $u \in U$, as defined by z [19].

Linguistic variable. The operation of fuzzy expert systems is based on the concept of a linguistic variable [15]. Each linguistic variable has a set of values, which are fuzzy variables forming its term set [6, 14].

Linguistic Variable L is characterized by the following set of properties:

$$L = (X, T(X), U, G, M), \tag{6.1}$$

where X – the name of the variable; $T(X)$ – the term set of variable X, which is a set of linguistic values for X, each value is a fuzzy variable (x') with values from the universal set U associated with a base variable u; G – the syntactic rule that generates the names of the values of variable X; M – the semantic rule that assigns each fuzzy variable x' its meaning $M(x')$, which is a fuzzy subset $M(x')$ of the universal set U.

Fuzzy rule base of an expert system. The behavior of the studied system is described in a limited natural language using linguistic variables [17, 18]. The input and output parameters of the system are treated as linguistic variables, and the process is described by a set of rules [9, 10]. The formal model of the rule base for the developed expert system is represented as follows [11, 12]:

$$\begin{aligned} R_1 &: A_{1,1} \text{ and/or } A_{1,2} \text{ and/or } \ldots \text{ and/or } A_{1,m1} \rightarrow B_{1,1} \text{ and/or } \ldots \text{ and/or } B_{1,k1}, \\ R_2 &: A_{2,1} \text{ and/or } A_{2,2} \text{ and/or } \ldots \text{ and/or } A_{2,m2} \rightarrow B_{2,1} \text{ and/or } \ldots \text{ and/or } B_{2,k2}, \\ R_n &: A_{n,1} \text{ and/or } A_{n,2} \text{ and/or } \ldots \text{ and/or } A_{n,mn} \rightarrow B_{n,1} \text{ and/or } \ldots \text{ and/or } B_{n,kn}, \end{aligned} \tag{6.2}$$

where $A_{i,j}$, $i = 1,2,\ldots,n$, $j = 1,2,\ldots,m_i$ – fuzzy statements defined on the values of input linguistic variables; $B_{i,q}$, $i = 1,2,\ldots,n$, $q = 1,2,\ldots,k_i$ – fuzzy statements defined on the values of output linguistic variables.

In general, fuzzy reasoning is carried out in four stages [13–15]:

1. Fuzzification stage: conversion of precise input data into fuzzy values of linguistic variables using membership functions.

2. Fuzzy inference stage: based on the set of rules in the fuzzy knowledge base, the truth values for the conditions of each rule are calculated using T-norm, T-conorm, and negation operations.

3. Composition stage: values of output linguistic variables are formed for each rule that has been triggered.

4. Defuzzification stage: conversion of fuzzy values of output linguistic variables into precise values.

Fuzzy logical inference. Let examine the stages of fuzzy decision-making in detail [19]:

1. *Fuzzification Stage*. Using the membership functions of all terms for input linguistic variables and based on precise input values from the universes of input linguistic variables, the degrees of confidence are determined. These represent the likelihood that the output linguistic variable assumes specific values [11, 12].

2. *Fuzzy Inference Stage*. From the set of rules (fuzzy knowledge base), the truth value for each rule is calculated based on specific fuzzy operations corresponding to the conjunction or disjunction of terms in the left-hand side of the rules. Typically, the maximum or minimum of the confidence degrees of terms, determined during fuzzification, is applied to the conclusion of each rule. Using one of the methods for constructing fuzzy implication, a fuzzy variable is generated that

corresponds to the computed confidence degree in the left-hand side of the rule and the fuzzy set in the right-hand side [13, 14].

3. *Composition (Aggregation or Accumulation) Stage*. All fuzzy sets assigned to each term of each output linguistic variable are combined into a single fuzzy set, representing the value of each output linguistic variable derived. This is typically achieved using maximum or summation functions [15, 16].

4. *Defuzzification Stage.* Defuzzification is applied when it is necessary to transform a fuzzy set of derived linguistic variable values into precise ones. There are numerous methods for this transition, with the most commonly used being the full interpretation method and the maximum interpretation method.

Full Interpretation Method: the precise value of the output variable is computed as the "center of gravity" of the membership function for the fuzzy value.

Maximum Interpretation Method: the precise value of the output variable is determined as the maximum value of the membership function [17–19].

The highest computational costs are incurred during the fuzzy inference stage. To address this, the study examines a proposed method for accelerating the decision-making process at this stage [10, 11].

Task of accelerating decision-making in fuzzy expert systems. To formulate the task of accelerating decision-making, the concept of a single iteration of fuzzy inference is introduced [12, 13]. It is proposed to represent it as a function F, which transforms a set of conditions into a set of consequences, as follows:

$$F:\{A_{1,1},A_{1,2},\ldots,A_{1,m1},A_{i,1},A_{i,2},\ldots,A_{i,mi},A_{n,1},A_{n,2},\ldots,A_{n,mn}\}\rightarrow$$
$$\rightarrow\left\{B_{1,1},B_{1,2},\ldots,B_{1,k1},B_{i,1},B_{i,2},\ldots,B_{i,ki},B_{n,1},B_{n,2},\ldots,B_{n,kn}\right\}. \quad (6.3)$$

The task of accelerating decision-making is to minimize the computations performed during the processing of the condition matrix A for fuzzy rules, this involves constructing a reduced set of fuzzy conditions A^*, that $|A^*|<|A|$, ensuring that the result remains consistent. Specifically, if $F(A)\rightarrow B$, then $F(A^*)\rightarrow B$.

Acceleration of decision-making can be achieved in two ways [14–16]:

1. Exclusion of certain rules from processing.

Suppose certain rules are excluded $i1, i2,\ldots, is$. In this case:

$$|A^*|=|A|-\sum_{t=1}^{S}(m_{it}). \quad (6.4)$$

2. Identifying identical conditions and eliminating their redundant computation: assume that p matches of conditions of the form $A_{i,j}=A_{v,w}$, where $i=1,2,\ldots,n$, $j=1,2,\ldots,m_n$, $v=1,2,\ldots,n$, $w=1,2,\ldots,m_n$. Where:

$$|A^*|=|A|-p. \quad (6.5)$$

The input data for the proposed method are the rules from the fuzzy rule base.

The following has been established:

– the methods (models, approaches) for knowledge representation in intelligent decision support systems presented in the study, in their canonical form, are not feasible for use due to a number of objective reasons outlined in Section 3 of the study;

– it is necessary to develop new (or improve existing) methods of knowledge representation in intelligent decision support systems that combine the advantages of these approaches without their disadvantages.

Future research should focus on further refinement of these approaches to reduce the number of drawbacks and limitations in their application.

6.2 DEVELOPMENT OF AN EVALUATION AND PREDICTION METHODOLOGY USING FUZZY COGNITIVE MAPS

The methodology for evaluation and prediction using fuzzy cognitive maps includes the following interrelated procedures:

1. *Input of Initial Data on the Object's State:* collecting the primary data about the object's condition.

2. *Initialization of the Object's Initial State Model:* setting up the base model to reflect the initial state of the object.

3. *Introduction of Corrective Coefficients for Noise and A Priori Uncertainty:* adjusting for noise and uncertainty using specific expressions [2].

Due to the absence of prior information about coefficients and the order of the differential equation, binary representation of optimization variables becomes challenging and inefficient for finding solutions. However, when information about data noise and uncertainty levels regarding the object's state is available, it becomes possible to improve the accuracy of constructing fuzzy cognitive maps.

4. *Construction of a fuzzy cognitive map of the object's state.*

The transition from a vector (individual) to a differential equation must consider the order, structure, and coefficients of the equation, as these influence the optimization algorithm's performance [11–15].

To construct a cognitive map reflecting the dynamic properties of the situation, scales for factor values and their increments must be defined.

The scale of a factor is determined by structuring the set of its linguistic values. Absolute values of factors, rather than subjective assessments like "large", "medium", or "small" are used to define linguistic values. An objective reference value (benchmark) for the factor is established as the standard for its value. This reference simplifies the expert's work in assessing the impact strength of factors and reduces errors.

The prediction involves the max-triangulation composition of the weight matrix and the vector of initial factor increments.

This algorithm is designed for positive-definite matrices. However, in this case, elements of the adjacency matrix and increment vectors may take both negative and positive values.

The following rule is used for transforming the adjacency matrix: $W = \left| w_{ij}sl \right| n \times n$ with positive and negative elements into a positively defined double matrix $W = \left| w'_{ij}sl \right| 2n \times 2n$:

$$\text{if } \left(w_{ij}sl\right) > 0, \text{ then } w'_{i(2j-1)}s_{(2l-1)} = w_{ij}sl,\ w'_i\left(2_j\right)s\left(2l\right) = w_{ij}sl; \tag{6.6}$$

$$\text{if } \left(w_{ij}sl\right) < 0, \text{ then } w'_{i(2j)}s_{(2l)} = -w_{ij}sl,\ w'_i\left(2_j\right)s\left(2l-1\right) = -w_{ij}sl. \tag{6.7}$$

Initial increment vector $P(t)$ and the vector of predicted feature values $P(t+1)$ in this case must have a dimensionality of $2n$. The rule for obtaining the initial increment vector $P'(t)$ of dimensionality $2n$ from the initial vector $P(t)$ of dimensionality n is as follows:

$$\text{if } p_{ij}\left(t\right) > 0, \text{ then } p'_i\left(2_j - 1\right)\left(t\right) = p_{ij}\left(t\right),\ p'_i\left(2_j\right)\left(t\right) = 0; \tag{6.8}$$

$$\text{if } p_{i_j}\left(t\right) < 0, \text{ then } p'_i\left(2_j\right)\left(t\right) = p_{ij}\left(t\right),\ p'_i\left(2_j - 1\right)\left(t\right) = 0. \tag{6.9}$$

In the vector $P'\left(t\right) = \left(p_{11}-, p_{11}+, \ldots, p_{nm}-, p_{nm}+\right)$ the value of the feature f_{ij} characterizes two elements: the element with index 2_j characterizes the positive $p_{ij}+$, and the element with index $2_j - 1$ – negative p_{ij} – the increment of the feature f_{ij}. Then the double increment vector $P'\left(t+1\right)$ for the positively defined matrix W' is determined using the following equation:

$$P'\left(t+1\right) \ = P'\left(t\right) \circ W', \tag{6.10}$$

where to calculate the element of the vector $P'\left(t+1\right)$ the following rule is used:

$$p'_{ij}\left(t+1\right) = \max_{sl}\left(p'sl\left(t\right) \cdot w'_{ij}sl\right). \tag{6.11}$$

The elements of the increment vectors of feature values, obtained at successive moments in time $P'\left(t+1\right),\ldots,P'\left(t+n\right)$, after transposition, are represented as a block matrix:

$$P_t = \left| P'\left(t+1\right)T, \ldots, P'\left(t+n\right)T \right|. \tag{6.12}$$

The rows of this matrix represent the increments of a single feature over successive moments in time, while the columns represent the increments of all features at the moment corresponding to the selected column. The matrix P_t is called the increment matrix and is used in the operation of algorithms for explaining the predictions of situation development [16–18].

5. *Forecasting the dynamics of an object's state.*

A set of situation factors $F = \{f_j\}$, $j = 1,\ldots,m$; $Z_j = \{z_{jk}\}$ – given as an ordered set of linguistic values for the i-th factor, k – denotes the index of the linguistic value, and the scales for all factors are defined X_i.

The cognitive map is determined by experts (F, W), where F – a set of vertices representing the situation factors, $W = |w_{ij}|$ – an adjacency matrix, the initial state of the situation is represented as a vector containing the values of all situation factors $X(0) = (x_{10},\ldots,x_{m0})$. The initial increment vector of the situation factors is $P(t) = (p_1,\ldots,p_m)$.

It is necessary to find the state vectors of the situation $X(t), X(t+1),\ldots,X(t+n)$ and the increment vectors of the situation's state $P(t), P(t+1),\ldots,P(t+n)$ at successive discrete moments in time t, $t+1$, …, $t+n$, where t – the step number (iteration) of the modeling process.

The forecast of situation development is determined using the matrix equation:

$$P(t+1) = P(t)^{\circ}W,$$

where (°) – rule *max-product*: $p_i(t+1) = \max_j(p_j(t)w_{ij})$.

An element of the forecast vector for situation development $p_i(t+1) \in P(t+1)$ is represented as a pair: $\langle p_i(t+1), c_i(t+1)\rangle$, where $p_i(t+1)$ is the increment value of the factor; $c_i(t+1)$ is the consonance of the factor value. Cognitive consonance characterizes the subject's confidence in the modeling results. When $c_i(t) \approx 1$ the subject's confidence in the increment of factor $p_i(t)$ is maximal, and when $c_i(t) \approx 0$ is minimal.

The state of the situation at successive time points is represented as a pair: $\langle X(t+1), C(t+1)\rangle$, where $X(t+1) = X(t) + P(t+1)$ is the state vector of the situation (with element $x_i(t+1) = x_i(t) + p_i(t+1)$), the cognitive consonance of the factor values $c_i(t+1) \in C(t+1)$.

A plausible forecast of situation development in this case is defined as a pair $\langle X(m), C(m)\rangle$, where $X(m) = (x_1(m),\ldots,x_m(m))$ is the vector of situation factor values at time $t = m$; $C(m) = (c1(m),\ldots,cm(m))$ is the vector of consonance values for the situation factors at time $t = m$.

6. *Training a fuzzy cognitive map using a genetic algorithm.*

Suppose there is a set of $3N$ rows of historical data (referred to as the training material) describing the state of concepts in the system. From the perspective of forecasting based on concept increments, the increments of concepts from the i-th iteration to $k(i+1)$ iteration form the output increment vector. In this case, the fuzzy cognitive map should indicate that, given a similar output increment vector, the concept values will change in such a way that their increments result in values at the $(i+2)$ iteration.

Let $A_i(t)$ represents the value of a concept at time t. Based on the specification of the training material described above, let's consider triples of rows: $A_i(t)$, $A_i(t+1)$, $A_i(t+2)$.

Define $x_i = \dfrac{A_i(t+1) - A_i(t)}{A_i(t)}$, $y_i = \dfrac{A_i(t+2) - A_i(t)}{A_i(t)}$, x – the output increment vectors, y – the resulting increment vectors.

To solve the training task, a genetic algorithm is proposed. A chromosome is represented as a one-dimensional array of values corresponding to the two-dimensional weight matrix of the fuzzy cognitive map. Each value in this array is called a gene. The main steps of the algorithm are:

1. For all non-zero weights of the initial map, a new non-zero weight value is assigned as a small random value. The initial non-zero weights are determined by the expert (any non-zero value can be assigned; its sole purpose is to indicate that, in the expert's opinion, a causal relationship exists between the two selected concepts).

2. Step 1 is repeated *Population Size* times. Thus, an initial population of Random solutions are formed.

3. The fitness function is calculated for each chromosome (the type of fitness function will be discussed later).

4. A pool of parents is determined using the "roulette wheel" selection method.

5. "Elite individuals" are added to the parent pool. In genetic algorithms, elite individuals are those that have shown the best fitness values over the last few generations (one elite individual from each generation).

6. Crossover occurs among chromosomes in the parent pool. The crossover between chromosomes *A* and *B* is performed as follows. The crossover point is determined randomly. Let's denote it as A_{l+} a segment of the chromosome *A*, consisting of genes located starting from *l*, and A_{l-} – the segment of the chromosome located before *l*. Then the result of the crossover will be two chromosomes: $A_{l-}B_{l+}$ and $B_{l-}A_{l+}$. The probability of crossover is predefined. If crossover does not occur, both parent chromosomes are transferred unchanged to the offspring population.

7. A new population is formed from the offspring obtained in Step 6. The size of this population is identical to the size of the population in the previous stage of the algorithm.

8. Mutations occur in the offspring population. During mutation, a random gene is selected and replaced with a new random value. The probability of mutation is predefined. If no mutation occurs, the chromosome proceeds to the next iteration of the algorithm unchanged.

9. The following parameters for the generation are determined: elite individual (the individual with the highest fitness value) retained to preserve its genetic makeup; average fitness value (this is calculated for the population and is used only to evaluate the convergence of the algorithm); fitness value of the elite individual.

10. If the fitness value of the elite individual exceeds a predefined maximum fitness value, the algorithm stops, and the selected chromosome is decomposed into the adjacency matrix of the fuzzy cognitive map (training is considered complete). Otherwise, the algorithm returns to Step 3.

End.

A methodology for evaluation and prediction using fuzzy cognitive maps has been developed. The concept of elite individuals was introduced into the algorithm to accelerate its convergence. The number of elite individuals was set to 60, while the population size was 100. Thus, after the 60^{th} generation, only 40 chromosomes from the current population have a chance to undergo crossover, while the rest populate the elite gene pool inherited from previous populations.

This high mutation probability, which is typically uncharacteristic of genetic algorithms, is justified in this case as it introduces genetic diversity into the population. Furthermore, the use of an elite gene pool mitigates the risk of irrevocably losing beneficial genes from previous generations.

The transient processes of different systems may converge over a specific interval. Therefore, increasing the observation interval for the system's output and the measurement frequency can improve the effectiveness of finding a solution. On the other hand, this may be attributed to the presence of numerous local optima and a relatively strong attraction zone.

The proposed methodology, unlike existing ones:

– accounts for the degree of uncertainty in information about the state of a dynamic object and the noise in the output data regarding its state;

– creates a multilevel and interconnected description of hierarchical objects;

– enhances decision-making speed in assessing object states by searching for solutions using population individuals;

– addresses the issue of reaching a global optimum.

Advantages of this research include:

– consideration of the degree of uncertainty about the object's state during calculations;

– accounting for data noise resulting from distortion in the information about the object's state;

– reduction in computational costs when evaluating object states;

– the ability to perform calculations with input data of different natures and units of measurement.

Disadvantages of this research include: the need for significant computational resources and time to perform calculations.

The proposed methodology is advisable for implementation in specialized software used for analyzing the states of complex technical systems and making management decisions.

Future research should focus on further improving this methodology to account for a larger number of factors during state analysis.

6.3 DEVELOPMENT OF A METHOD FOR COGNITIVE REPRESENTATION OF THE STATE OF COMPLEX HIERARCHICAL SYSTEMS

Currently, there are no unified principles for constructing cognitive representations that convey sufficient information in a concise and user-accessible form to make appropriate management decisions [1–3].

Typically, representations are created individually, considering the specific applied domain and interpreted by an expert (or a group of experts) based on accumulated knowledge. Multidimensional data can be transformed into cognitive graphical representations using computational tools in the form of integral functional profiles, which reflect the states of complex hierarchical systems [3–6].

Existing mathematical methods for analyzing and visualizing multidimensional data are poorly suited to real-time dynamic systems and lack sufficient versatility, hindering their widespread adoption.

The aim of the study is to develop a method for cognitive representation of the states of complex hierarchical systems with the following features [4–9]:

– the systems operate in real-time;

– various scales and ranges of controlled parameter changes are applied;

– data stream interruptions and system failures may occur [2].

To visualize the current state in such systems, a cognitive graphical representation is required that meets the following requirements [3, 4]:

1. A mathematical framework for transforming feature space into image space:

– expressiveness of images to accelerate experts' understanding of the current situation;

– unambiguous and accurate representation of situation classes: "normal", "anomaly", "critical", "no info" (absence of information);

– capability to represent the state of complex hierarchical systems as a whole and the states of their individual subsystems at all hierarchy levels.

2. Ability to display parameters with an indication of the deviation level from the average within an acceptable operating range.

3. A unified formalism for describing relationships important for decision-making in high-dimensional symbolic space.

Considering the methods for constructing cognitive representations, multilevel approaches to presenting situations are the most effective for monitoring hierarchical systems. In practical applications, the number of levels typically does not exceed three, as increasing them complicates perception and reduces the efficiency of state analysis. The results of the review of methods for representing the states of complex hierarchical objects are summarized in **Table 6.1** [10–19].

Analysis of **Table 6.1** shows that the cognitive representation of information using fractals best meets the formulated requirements. However, it has several disadvantages that reduce its ergonomic qualities, namely: a large number of small details, a low permissible embedding depth, and the absence of numerical information on the cognitive image slide.

As a basis for transforming the parameter space into the space of graphical representations, the method of integral contour representation of a standard epicycloid is adopted:

$$\begin{cases} x = (R+d)\cos\varphi - d\cos((m+1)\varphi; \\ y = (R+d)\sin\varphi - d\sin((m+1)\varphi, \end{cases} \quad (6.13)$$

and standard hypocycloids:

$$\begin{cases} x = (R-d)\cos\varphi + d\cos((m-1)\varphi; \\ y = (R-d)\sin\varphi - d\sin((m-1)\varphi, \end{cases} \quad (6.14)$$

where R is the radius of the stationary circle; r is the radius of the rolling circle; $d=r$ is the distance of point M from the center C of the rolling circle; φ is the parameter describing the angle between the line segment connecting the circle centers and the OX; m is an integer defined a, $m=R/r$.

Table 6.1 Overview of methods for representing the state of complex hierarchical objects

No.	Name	Purpose	Types of systems		Deviation level	Universality
			Dynamic	Complex Hierarchies		
1.	*n*-Simplex	Pattern recognition	–	+	+	+
2.	Meta	Pattern recognition	–	+	+	–
3.	Star	Graphical representation of multidimensional data	+	–	–	+
4.	Botanical tree	Visualization of complex hierarchies	–	+	–	–
5.	Novosyolov fractal	Diagnostics and visualization for complex technical objects	+	+	–	+
6.	Image for nuclear power plant operators	Display of anomalies and current situations in complex systems	+	+	+	–
7.	Color coding based on evaluation function	Assessment of object state compliance at a given time	+	–	+	–
8.	Large-scale electrical network image	Diagnostics and visualization for complex technical objects	+	+	+	–
9.	Drum-separator image	Decision support for nuclear power plant operators	+	–	+	–
10.	Shoke integral	Selection of constraints on parameters of fuzzy aggregation operators for interrelated criteria	–	–	+	–
11.	Fuzzy cognitive maps	Analyst decision support	–	–	+	–

When the rolling circle rotates around the stationary circle $O(x_0, y_0)$, by an angle multiple of $2\pi r/R$, the epicycloid (or hypocycloid) overlaps with itself. Both the epicycloid and hypocycloid consist of m congruent branches. Let j – the branch number, and if $j = 1,\ldots,m$, then the parametric equations for the j-th congruent branch of the epicycloid are derived from expressions (6.13) and (6.14):

$$\begin{cases} x = x_0 + (R+d)\cos\psi - d\cos(m+1)\psi; \\ y = y_0 + (R+d)\sin\psi - d\sin(m+1)\psi, \end{cases} \tag{6.15}$$

and for the hypocycloid:

$$\begin{cases} x = x_0 + (R-d)\cos\psi + d\cos(m-1)\psi; \\ y = y_0 + (R-d)\sin\psi - d\sin(m-1)\psi, \end{cases} \tag{6.16}$$

where x_0 and y_0 – the coordinates of the center of the stationary circle, and $\psi \in \left[j\alpha, (j+1)\alpha \right]$, $\alpha = 2\pi/m$. From equations (6.15) and (6.16), let's obtain the generalized formula for the branches of the epicycloid and hypocycloid:

$$\begin{cases} x = x_0 + (R + \xi d)\cos\psi - \xi d \cos(m + 1\xi)\psi; \\ y = y_0 + (R + \xi d)\sin\psi - d \sin(m + 1\xi)\psi, \end{cases} \tag{6.17}$$

where $\xi = 1$ for the epicycloid and $\xi = -1$ for the hypocycloid.

Let the hierarchical system be characterized by a set of parameters $Z = \{z_1, z_2, \ldots, z_j, \ldots, z_m\}$. It is proposed to use formula (6.17) as the basis for integral contour representation to address the problem of detecting parameter values z_j that exceed the permissible range $\left[z_{j_{\min}}, z_{j_{\max}} \right]$. Let d equal the value of parameter z_j, normalized to the interval $\left[-\delta, \delta \right]$. For normalization, let's use the formula:

$$\overline{z}_j = \delta \left(2 \left(\frac{z_j - z_{j_{\min}}}{z_{j_{\max}} - z_{j_{\min}}} \right) - 1 \right), \tag{6.18}$$

where $\delta = R/2\Phi^2$, Φ is the golden ratio coefficient.

If the parameter z_j equals the mean value within the permissible range, then $d = \overline{z}_j = 0$ and the curve described by (6.17) becomes a part of a circle. If $\overline{z}_j < 0$, the curve (6.17) forms a congruent branch of a hypocycloid. If $\overline{z}_j > 0$, the curve (6.5) forms a congruent branch of an epicycloid.

Substituting in formula (6.17) ξd on $\overline{z}_j$, d on $\left|\overline{z}_j\right|$, the type of the curve (6.17) will be determined by the deviation of the controlled parameter from the mean value within the permissible range:

$$\begin{cases} x = x_0 + (R + \overline{z}_j)\cos\psi - \overline{z}_j \cos(m + \eta)\psi; \\ y = y_0 + (R + \overline{z}_j)\sin\psi - \left|\overline{z}_j\right| \sin(m + \eta)\psi, \end{cases} \tag{6.18}$$

where $\eta = \begin{cases} 1, & \text{if} \quad \overline{z}_j > 0; \\ -1, & \text{if} \quad \overline{z}_j < 0; \\ 0, & \text{if} \quad \overline{z}_j = 0. \end{cases}$

In cases where $z_j = q_1 z_{j_{\min}}$, $q_1 > 1$ or when $z_j = q_2 z_{j_{\max}}$, $q_2 > 1$, $\{q_1, q_2\} \in R$, the d, the corresponding $\left|\overline{z}_j\right|$, will exceed r, and the curve (6.18) will form a branch of an elongated epicycloid or hypocycloid, resulting in "loops". To eliminate loops, replace the epicycloid branches with elliptical arcs if $\overline{z}_j > r$, replace the hypocycloid branches with hyperbolic branches if $\overline{z}_j < -r$. The formula (6.18) is adjusted based on these refinements:

$$\begin{cases} x = x_0 + \left(R + \overline{z}_j\right)\cos\psi - \overline{z}_j \cos\left(m + \eta\right)\psi; \\ y = y_0 + \left(R + \overline{z}_j\right)\sin\psi - \left|\overline{z}_j\right|\sin\left(m + \eta\right)\psi, \end{cases}$$

if $-r \le \overline{z}_j \le r$;

$$\begin{cases} x = x_0^e + a^e \cos\beta\cos\tau - b^e \sin\beta\sin\tau; \\ y = y_0^e + a^e \cos\beta\sin\tau + b^e \sin\beta\cos\tau, \end{cases}$$

if $\overline{z}_j > r$;

$$\begin{cases} x = x_0 + a^h \cosh\theta\cos\tau - b^h \sinh\theta\sin\tau; \\ y = y_0 + a^h \cosh\theta\sin\tau + b^h \sinh\theta\cos\tau, \end{cases}$$

if $\overline{z}_j < -r$, (6.19)

where $\tau = j\alpha + \alpha/2$ is the angle of rotation of the ellipse or hyperbola around the point $O\left(x_0, y_0\right)$; $x_0^e = x_0 + R\cos\alpha/2\cos\tau$, $y_0^e = y_0 + R\cos\alpha/2\cos\tau$, $a^e = 2\overline{z}_j + R - R\cos\alpha/2$ is the major semi-axis of the ellipse; $b^e = R\sin\alpha/2$ is the minor semi-axis of the ellipse; $a^h = R + 2\overline{z}_j$ is the real semi-axis of the hyperbola; $b^h = \dfrac{a^h R\sin\dfrac{\alpha}{2}}{\sqrt{\left(R\cos\dfrac{\alpha}{2}\right)^2 - \left(a^h\right)^2}}$ is the imaginary semi-axis of the hyperbola, $\theta \in \left[-2\pi, 2\pi\right]$, $\beta \in \left[-\pi/2, \pi/2\right]$.

The proposed method, unlike existing ones:

– creates a visual, multilevel, and interconnected description of a hierarchical system;

– improves the efficiency of decision-making in assessing the state of a hierarchical system;

– resolves the issue of global and local extrema when evaluating the state of a hierarchical system;

– combines graphical and numerical representations of controlled parameters for the state of the national security system;

– enables real-time visualization of the state of a hierarchical system;

– avoids the issue of loop formation during the real-time visualization of the state of a hierarchical system;

– ensures the accuracy of the hierarchical system state visualization, regardless of the number of individual components comprising the system.

Advantages of this study include:

– the ability to perform calculations with input data of different natures and measurement units;

– the ability to avoid loop formation during the visualization of the hierarchical system state;

– the creation of a visual, numerical, multilevel, and interconnected description of the hierarchical system.

Disadvantages of this study include the need for appropriate computational resources and time to perform calculations.

The proposed method is recommended for implementation in specialized software used for analyzing the state of hierarchical systems and making managerial decisions.

6.4 DEVELOPMENT OF A METHOD FOR EVALUATING COMPLEX HIERARCHICAL SYSTEMS BASED ON AN ENHANCED PARTICLE SWARM OPTIMIZATION

This study proposes a method for evaluating complex hierarchical systems using an enhanced particle swarm optimization (PSO) combined with the coordinate averaging method.

This approach allows the coordinate averaging method to shift from the random selection of trial points to utilizing the current coordinates of the particle swarm, whose collective movement is adaptive, adjusting to the characteristics of the objective function's changes. During the movement of the particle swarm, their displacement is adjusted toward the determined averaged center based on the coordinate averaging method. An additional mechanism that accelerates the convergence of the hybrid algorithm is the inclusion of several steps of the Hooke-Jeeves procedure. These steps refine the current coordinates of the best and/or worst particle in the swarm.

This study focuses primarily on "zero-order" methods, in which the values of the objective function are determined only at trial points through a computational algorithm. This approach is oriented towards solving tasks for determining the objective function of a complex dynamic process with minimal requirements for the continuity and boundedness of the objective function. For approximate estimates of the variability of the objective function, the method utilizes the maximum values obtained as the ratio of the difference in the objective function values to the distance between all pairs of trial points (a lower bound for the Lipschitz constant).

A bounded continuous function is considered $f(x): \Omega \to \mathbb{R}$, where $x = (x_1, x_2, \ldots, x_n) \in \Omega \subset \mathbb{R}^n$. The set Ω represents the domain of allowable variable values and, in the simplest case, is an n-dimensional parallelepiped with given sides, $\left[x_i^{[0]} - d_i, x_i^{[0]} + d_i\right]$, $i = 1,2,\ldots,n$.

The task is to find an approximate value of the global minimum f^* and at least one point x^*, where this value is achieved, with a specified permissible ε^f for the objective function values:

$$f_{\min} = \min f(x),$$
$$x \in \Omega, f^* - \varepsilon_f \le f_{\min} \le f^*, f^* = f(x^*). \tag{6.20}$$

The computational procedure for finding the approximate coordinates of the point x^* in the coordinate averaging method is based on an iterative process, which in its continuous form is expressed as [5]:

$$x_i^{[k+1]} = \left(\int_{\Omega^{[k]}} x_i p_s^{[k]}(x) dx \right) \times \iota, \; i = 1,2,\ldots,n. \tag{6.21}$$

$$p_s^{[k]}(x) = \frac{P_s^{[k]}\left(f(x)\right)}{\int\limits_{\Omega^{[k]}} P_s^{[k]}\left(f(x)\right)\mathrm{d}x}, \tag{6.22}$$

where k is the step number of the computational process; ι is the degree of uncertainty in the state of a complex real-time dynamic system (values range from 0 to 1); $\Omega^{[k]}$ is the coordinate averaging region at step k. A sequence of continuous functions $P_s(y)$, $s=1,2,3,\ldots$, such that $\forall y \in \mathbb{R}$ and value $P_s(y) \geq 0$ and for the sequence $P_s(y)/P_z(y)$ the condition of monotonic unbounded growth holds with increasing selectivity parameter s for any fixed values y, z with the condition $y < z$. Examples of such functions include $P_s(y)$, in particular, are functions $\exp(-sy)$, s^{-y}, y^{-s}, as well as a class of functions of the form $(1-y^r)^s$ for $y \in [0,1]$, $r=1,2,3,\ldots$. For the numerical minimization examples below, the function is used $(1-y^2)^s$.

As increases s the steepness of the kernels $p_s^{[k]}$ grows, which in turn increases the weights of the coordinates corresponding to better values of the objective function (OF). In the final case, the sequence of averaged coordinates converges to the global minimum (the corresponding convergence theorem is proven in [5]).

For the numerical implementation of the coordinate averaging method, one effective way to improve the accuracy of integral computations is to sequentially increase the number of trial points $x^{(j)[k]}$, $j = 1,2,\ldots,M^{[k]}$, at the k-th stage of the iterative process, i.e. $M^{[k]} \geq M^{[k-1]}$. To avoid accidentally excluding the global minimum point, the averaging region $\Omega^{[k]}$ in this case can be considered either adaptively variable or fixed [5].

In the proposed modification of the iterative coordinate averaging algorithm, adaptive displacement of trial points is introduced. This is implemented as the movement of particles in the PSO method with an FDR (fitness-distance ratio-based PSO) modification [22]. Additionally, particles in the swarm are shifted toward the determined center of averaged coordinates, introducing a new factor of information exchange between particles and adding extra stabilization to the collective swarm search for the global minimum of the objective function (OF).

When calculating integrals in formulas (6.21) and (6.22), the summation of the values of the integrand expressions is performed over the set of trial points, taking into account the volumes of the subregions discretizing the integration domain $\Omega^{[k]}$.

In the proposed hybrid algorithm, the coordinates of the trial points are identified with the coordinates of the particle swarm, which change during the collective search for the global minimum. At the beginning of the computational process, these coordinates are initialized either at the nodes of a computational grid or generated using a random number generator with a uniform distribution over the intervals $\left[x_i^{[0]} - d_i, x_i^{[0]} + d_i\right]$, $i = 1,2,\ldots,n$. In discrete form, the relationships (6.21) and (6.22) are expressed as follows:

$$x_i^{[k+1]} = \sum_{j=1}^{M^{[k]}} x_i^{(j)[k]} p_s^{[k]}\left(x^{(j)[k]}\right) V^{(j)[k]}, \; i = 1,2,\ldots,n, \tag{6.23}$$

$$p_s^{[k]}\left(x^{(j)[k]}\right)=\frac{P_s^{[k]}\left(g^{[k]}\left(x^{(j)[k]}\right)\right)}{\sum\limits_{j=1}^{M^{[k]}}P_s^{[k]}\left(g^{[k]}\left(x^{(j)[k]}\right)\right)V^{(j)[k]}}, \tag{6.24}$$

where $V^{(j)[k]}$ corresponds to the n-dimensional volume obtained by dividing the domain $\Omega^{[k]}$ into subdomains associated with the family of integration points $x^{(j)[k]}$, $j=1,2,\ldots,M^{[k]}$.

Here $g^{[k]}\left(x\right)$ are auxiliary functions that scale the objective function $f\left(x^{(j)[k]}\right)$ to the range [0, 1], defined as:

$$g^{[k]}\left(x^{(j)[k]}\right)=\frac{f\left(x^{(j)[k]}\right)-f^{[k]}_{\min}}{f^{[k]}_{\max}-f^{[k]}_{\min}}, \tag{6.25}$$

$$f^{[k]}_{\min}=\min\left(f\left(x^{(1)[k]}\right),f\left(x^{(2)[k]}\right),\ldots,f\left(x^{\left(M^{[k]}\right)[k]}\right)\right),$$

$$f^{[k]}_{\max}=\max\left(f\left(x^{(1)[k]}\right),f\left(x^{(2)[k]}\right),\ldots,f\left(x^{\left(M^{[k]}\right)[k]}\right)\right).$$

Assuming that for the trial points (current coordinates of the particle swarm) in formulas (6.24) and (6.25), during approximate integration, the subdomains can be chosen such that their sizes $V^{(j)[k]}$, $j=1,2,\ldots,M^{[k]}$, are approximately equal, the computational formulas simplify to the following form:

$$x_i^{[k+1]}=\sum_{j=1}^{M^{[k]}}x_i^{(j)[k]}p_s^{[k]}\left(x^{(j)[k]}\right),\ i=1,2,\ldots,n, \tag{6.26}$$

$$p_s^{[k]}\left(x^{(j)[k]}\right)=\frac{P_s^{[k]}\left(g^{[k]}\left(x^{(j)[k]}\right)\right)}{\sum\limits_{j=1}^{M^{[k]}}P_s^{[k]}\left(g^{[k]}\left(x^{(j)[k]}\right)\right)}. \tag{6.27}$$

It should be noted that for the coordinate averaging computational algorithm [5], the specific type of subdomains discretizing the integration domain $\Omega^{[k]}$ is not critical, thus, using the simpler relationships (6.26) and (6.27) instead of (6.24) and (6.25) for numerical implementation is entirely justified.

The adaptive adjustment of particle swarm coordinates at the $(k+1)$-th step follows the PSO scheme [10–18], with an additional component introduced for movement toward the averaged coordinate center $x^{[k]}$ for each j-th particle in the swarm, expressed as:

$$x^{(j)[k+1]}=x^{(j)[k]}+\alpha Y^{(j)[k]}+D^{(j)[k]}+U^{[0,\,\beta_0]}\otimes\left(x^{[k]}-x^{(j)[k]}\right), \tag{6.28}$$

where $Y^{(j)[k]}$ – the inertia component of the movement of the j-th particle; $D^{(j)[k]}$ – the adaptive displacement vector of the j-th particle, which is determined by three components of the random displacement of this particle [18]:

$$D^{(j)[k]} = d_1^{(j)[k]} + d_2^{(j)[k]} + d_3^{(j)[k]}, \tag{6.29}$$

where

$$\begin{aligned} d_1^{(j)[k]} &= U\left[0,\beta_1\right] \otimes \left(x_b^{(j)[k]} - x^{(j)[k]}\right), \\ d_2^{(j)[k]} &= U\left[0,\beta_2\right] \otimes \left(x_g^{(j)[k]} - x^{(j)[k]}\right), \\ d_3^{(j)[k]} &= U\left[0,\beta_3\right] \otimes \left(x^{(q(j))[k]} - x^{(j)[k]}\right). \end{aligned} \tag{6.30}$$

In formulas (6.28)–(6.30), the following notations are used:

– $x_b^{(j)[k]}$ – the best coordinates of the j-th particle over iterations, determined based on the objective function value ($d_1^{(j)[k]}$ – represents the cognitive component of the particle displacement);

– $x_g^{(j)[k]}$ – the coordinates of the best particle in the swarm with the minimum objective function value over k iterations ($d_2^{(j)[k]}$ – represents the social component of the particle displacement);

– $x^{(q(j))[k]}$ – coordinates of the particle with index $q(j)$, in the direction of which the objective function's rate of decrease is the greatest ($d_3^{(j)[k]}$ – a component of the objective function's variability based on the local Lipschitz constant estimate);

– $U\left[0,\beta\right]$ – a vector with components uniformly distributed random numbers in the interval $\left[0,\beta\right]$; $\otimes$ – element-wise multiplication of vectors; coefficients α, β_m, $m = 0,1,2,3$ that are tunable parameters of the hybrid computational algorithm.

Thus, the relationships (6.24)–(6.30), after specifying the type of kernels $P_s^{[k]}\left(y\right)$ with an increasing selectivity parameter and assigning specific values to the coefficients α, β_m, $m = 0,1,2,3$, fully define the hybrid computational algorithm for global optimization based on the coordinate averaging and particle swarm methods. The selection of coefficients for the numerical global optimization algorithm can be performed through meta-optimization [19], which is beyond the scope of this research.

The algorithm was run 100 times, and acceptable accuracy (on the order of 10^{-2}) in the coordinates of the best particle was achieved within 10–15 iterations, followed by the sequential concentration of particles near the global minimum. It should be noted that the method is statistical, and obtaining an "acceptable result with a certain probability" requires multiple runs of the software application with different random vector values $U\left[0,\ \beta\right]$.

Transitioning to higher dimensions of variables necessitates an exponential increase in the number of trial points or particles in the swarm. The possibility of parallel computations supports the feasibility of using multiple families of particle swarms in the algorithm.

Computational experiments minimizing function (6.29) with $n=100$ showed that if the total number of particles is not increased, the hybrid algorithm converges to one of the local minima. This occurs because the local minimum has the largest attraction basin (as reflected in the last term of formula (6.29)), which increases the likelihood that at least one particle enters this region, exerting the maximum influence on the swarm's subsequent behavior.

It is worth noting that the use of multiple families of particle swarms enables simultaneous identification of both the global minimum and local minima of the objective function in certain cases. This feature can be valuable for solving applied problems.

The proposed method, unlike existing ones:

– creates a multilevel and interconnected description of complex hierarchical real-time systems;

– improves decision-making efficiency in evaluating the state of complex hierarchical real-time systems;

– resolves the issue of reaching global and local extrema when evaluating the state of complex hierarchical real-time systems;

– enables directed search by multiple particles in a given direction, considering the degree of uncertainty;

– allows for repeated analysis of the state of complex hierarchical real-time systems.

Advantages of this study include:

– the ability to perform calculations with input data of different natures and units of measurement;

– the ability to conduct directed search by multiple particles in a given direction, considering the degree of uncertainty;

– the capability for repeated analysis of the state of complex hierarchical real-time systems.

Disadvantages of this study include the requirement for substantial computational resources and time to perform calculations.

This method is recommended for implementation in specialized software used for analyzing the state of complex hierarchical real-time systems.

CONCLUSIONS

1. An analysis of knowledge representation models was performed, and the advantages of applying production-based knowledge representation in expert systems were substantiated. The main concepts of fuzzy expert systems were outlined, based on which a formal problem statement for accelerating decision-making in the rule base of a fuzzy expert system was proposed. The stages of fuzzy logical inference were analyzed.

2. A methodology for evaluation and prediction using fuzzy cognitive maps was developed. The novelty of the proposed methodology lies in:

– accounting for a correction coefficient for the degree of uncertainty about the object's state in calculations;

– adding a correction coefficient for data noise resulting from distorted information about the object's state;

– reducing computational costs when evaluating object states;

– creating a multilevel and interconnected description of hierarchical objects;

– adjusting the object description due to changes in its current state using a genetic algorithm;

– enabling calculations with input data of different natures and measurement units.

This methodology is recommended for implementation in specialized software used for analyzing the states of complex technical systems and making management decisions.

3. A method for visualizing the states of hierarchical systems was developed. The novelty of the proposed method lies in:

– creating a visual, multilevel, and interconnected description of the hierarchical system;

– improving decision-making efficiency when evaluating the state of the hierarchical system;

– resolving the issue of reaching global and local extrema when evaluating the state of the hierarchical system;

– combining graphical and numerical representations of the controlled parameters of the hierarchical system's state;

– avoiding loop formation issues during real-time visualization of the hierarchical system's state.

4. A method for evaluating complex hierarchical systems based on enhanced particle swarm optimization (PSO) was developed. The proposed method is based on combining particle swarm optimization and coordinate averaging methods, along with modifications using multiple particle swarms and incorporating the Hooke-Jeeves procedure with corresponding correction coefficients. The novelty of the proposed method lies in:

– creating a multilevel and interconnected description of complex real-time hierarchical systems;

– improving decision-making efficiency when evaluating the states of complex real-time hierarchical systems;

– resolving the issue of reaching global and local extrema when evaluating the states of complex real-time hierarchical systems;

– enabling directed search by multiple swarm particles in a given direction, considering the degree of uncertainty;

– allowing repeated analysis of the states of complex real-time hierarchical systems.

REFERENCES

1. Dudnyk, V., Sinenko, Y., Matsyk, M., Demchenko, Y., Zhyvotovskyi, R., Repilo, I. et al. (2020). Development of a method for training artificial neural networks for intelligent decision support systems. Eastern-European Journal of Enterprise Technologies, 3 (2 (105)), 37–47. https://doi.org/10.15587/1729-4061.2020.203301

2. Sova, O., Shyshatskyi, A., Salnikova, O., Zhuk, O., Trotsko, O., Hrokholskyi, Y. (2021). Development of a method for assessment and forecasting of the radio electronic environment. EUREKA: Physics and Engineering, 4, 30–40. https://doi.org/10.21303/2461-4262.2021.001940
3. Pievtsov, H., Turinskyi, O., Zhyvotovskyi, R., Sova, O., Zvieriev, O., Lanetskii, B., Shyshatskyi, A. (2020). Development of an advanced method of finding solutions for neuro-fuzzy expert systems of analysis of the radioelectronic situation. EUREKA: Physics and Engineering, 4, 78–89. https://doi.org/10.21303/2461-4262.2020.001353
4. Zuiev, P., Zhyvotovskyi, R., Zvieriev, O., Hatsenko, S., Kuprii, V., Nakonechnyi, O. et al. (2020). Development of complex methodology of processing heterogeneous data in intelligent decision support systems. Eastern-European Journal of Enterprise Technologies, 4 (9 (106)), 14–23. https://doi.org/10.15587/1729-4061.2020.208554
5. Kuchuk, N., Mohammed, A. S., Shyshatskyi, A., Nalapko, O. (2019). The Method of Improving the Efficiency of Routes Selection in Networks of Connection with the Possibility of Self-Organization. International Journal of Advanced Trends in Computer Science and Engineering, 8 (1.2), 1–6.
6. Shyshatskyi, A., Zvieriev, O., Salnikova, O., Demchenko, Ye., Trotsko, O., Neroznak, Ye. (2020). Complex Methods of Processing Different Data in Intellectual Systems for Decision Support System. International Journal of Advanced Trends in Computer Science and Engineering, 9 (4), 5583–5590. https://doi.org/10.30534/ijatcse/2020/206942020
7. Pozna, C., Precup, R.-E., Horvath, E., Petriu, E. M. (2022). Hybrid Particle Filter-Particle Swarm Optimization Algorithm and Application to Fuzzy Controlled Servo Systems. IEEE Transactions on Fuzzy Systems, 30 (10), 4286–4297. https://doi.org/10.1109/tfuzz.2022.3146986
8. Yang, X.-S., Deb, S. (2013). Cuckoo search: recent advances and applications. Neural Computing and Applications, 24 (1), 169–174. https://doi.org/10.1007/s00521-013-1367-1
9. Mirjalili, S. (2015). The Ant Lion Optimizer. Advances in Engineering Software, 83, 80–98. https://doi.org/10.1016/j.advengsoft.2015.01.010
10. Yu, J. J. Q., Li, V. O. K. (2015). A social spider algorithm for global optimization. Applied Soft Computing, 30, 614–627. https://doi.org/10.1016/j.asoc.2015.02.014
11. Mirjalili, S., Mirjalili, S. M., Lewis, A. (2014). Grey Wolf Optimizer. Advances in Engineering Software, 69, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
12. Koval, V., Nechyporuk, O., Shyshatskyi, A., Nalapko, O., Shknai, O., Zhyvylo, Y. et al. (2023). Improvement of the optimization method based on the cat pack algorithm. Eastern-European Journal of Enterprise Technologies, 1 (9 (121)), 41–48. https://doi.org/10.15587/1729-4061.2023.273786
13. Gupta, E., Saxena, A. (2015). Robust generation control strategy based on Grey Wolf Optimizer. Journal of Electrical Systems, 11, 174–188.
14. Chaman-Motlagh, A. (2015). Superdefect Photonic Crystal Filter Optimization Using Grey Wolf Optimizer. IEEE Photonics Technology Letters, 27 (22), 2355–2358. https://doi.org/10.1109/lpt.2015.2464332

15. Nuaekaew, K., Artrit, P., Pholdee, N., Bureerat, S. (2017). Optimal reactive power dispatch problem using a two-archive multi-objective grey wolf optimizer. Expert Systems with Applications, 87, 79–89. https://doi.org/10.1016/j.eswa.2017.06.009
16. Koval, M., Sova, O., Orlov, O., Shyshatskyi, A., Artabaiev, Y., Shknai, O. et al. (2022). Improvement of complex resource management of special-purpose communication systems. Eastern-European Journal of Enterprise Technologies, 5 (9 (119)), 34–44. https://doi.org/10.15587/1729-4061.2022.266009
17. Ali, M., El-Hameed, M. A., Farahat, M. A. (2017). Effective parameters' identification for polymer electrolyte membrane fuel cell models using grey wolf optimizer. Renewable Energy, 111, 455–462. https://doi.org/10.1016/j.renene.2017.04.036
18. Zhang, S., Zhou, Y. (2017). Template matching using grey wolf optimizer with lateral inhibition. Optik, 130, 1229–1243. https://doi.org/10.1016/j.ijleo.2016.11.173
19. Khouni, S. E., Menacer, T. (2023). Nizar optimization algorithm: a novel metaheuristic algorithm for global optimization and engineering applications. The Journal of Supercomputing, 80 (3), 3229–3281. https://doi.org/10.1007/s11227-023-05579-4
20. Saremi, S., Mirjalili, S., Lewis, A. (2017). Grasshopper Optimisation Algorithm: Theory and application. Advances in Engineering Software, 105, 30–47. https://doi.org/10.1016/j.advengsoft.2017.01.004
21. Braik, M. S. (2021). Chameleon Swarm Algorithm: A bio-inspired optimizer for solving engineering design problems. Expert Systems with Applications, 174, 114685. https://doi.org/10.1016/j.eswa.2021.114685
22. Thamer, K. A., Sova, O., Shaposhnikova, O., Yashchenok, V., Stanovska, I., Shostak, S. et al. (2024). Development of a solution search method using a combined bio-inspired algorithm. Eastern-European Journal of Enterprise Technologies, 1 (4 (127)), 6–13. https://doi.org/10.15587/1729-4061.2024.298205
23. Yapici, H., Cetinkaya, N. (2019). A new meta-heuristic optimizer: Pathfinder algorithm. Applied Soft Computing, 78, 545–568. https://doi.org/10.1016/j.asoc.2019.03.012
24. Duan, H., Qiao, P. (2014). Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. International Journal of Intelligent Computing and Cybernetics, 7 (1), 24–37. https://doi.org/10.1108/ijicc-02-2014-0005
25. Shyshatskyi, A., Romanov, O., Shknai, O., Babenko, V., Koshlan, O., Pluhina, T. et al. (2023). Development of a solution search method using the improved emperor penguin algorithm. Eastern-European Journal of Enterprise Technologies, 6 (4 (126)), 6–13. https://doi.org/10.15587/1729-4061.2023.291008
26. Yang, X.-S. (2012). Flower Pollination Algorithm for Global Optimization. Unconventional Computation and Natural Computation, 240–249. https://doi.org/10.1007/978-3-642-32894-7_27
27. Gomes, G. F., da Cunha, S. S., Ancelotti, A. C. (2018). A sunflower optimization (SFO) algorithm applied to damage identification on laminated composite plates. Engineering with Computers, 35 (2), 619–626. https://doi.org/10.1007/s00366-018-0620-8

28. Mehrabian, A. R., Lucas, C. (2006). A novel numerical optimization algorithm inspired from weed colonization. Ecological Informatics, 1 (4), 355–366. https://doi.org/10.1016/j.ecoinf.2006.07.003
29. Qi, X., Zhu, Y., Chen, H., Zhang, D., Niu, B. (2013). An Idea Based on Plant Root Growth for Numerical Optimization. Intelligent Computing Theories and Technology. Berlin: Heidelberg, 571–578. https://doi.org/10.1007/978-3-642-39482-9_66
30. Bezuhlyi, V., Oliynyk, V., Romanenko, I., Zhuk, O., Kuzavkov, V., Borysov, O. et al. (2021). Development of object state estimation method in intelligent decision support systems. Eastern-European Journal of Enterprise Technologies, 5 (3 (113)), 54–64. https://doi.org/10.15587/1729-4061.2021.239854
31. Mahdi, Q. A., Shyshatskyi, A., Prokopenko, Y., Ivakhnenko, T., Kupriyenko, D., Golian, V. et al. (2021). Development of estimation and forecasting method in intelligent decision support systems. Eastern-European Journal of Enterprise Technologies, 3 (9 (111)), 51–62. https://doi.org/10.15587/1729-4061.2021.232718
32. Sova, O., Radzivilov, H., Shyshatskyi, A., Shevchenko, D., Molodetskyi, B., Stryhun, V. et al. (2022). Development of the method of increasing the efficiency of information transfer in the special purpose networks. Eastern-European Journal of Enterprise Technologies, 3 (4 (117)), 6–14. https://doi.org/10.15587/1729-4061.2022.259727
33. Zhang, H., Zhu, Y., Chen, H. (2013). Root growth model: a novel approach to numerical function optimization and simulation of plant root system. Soft Computing, 18 (3), 521–537. https://doi.org/10.1007/s00500-013-1073-z
34. Labbi, Y., Attous, D. B., Gabbar, H. A., Mahdad, B., Zidan, A. (2016). A new rooted tree optimization algorithm for economic dispatch with valve-point effect. International Journal of Electrical Power & Energy Systems, 79, 298–311. https://doi.org/10.1016/j.ijepes.2016.01.028
35. Murase, H. (2000). Finite element inverse analysis using a photosynthetic algorithm. Computers and Electronics in Agriculture, 29 (1-2), 115–123. https://doi.org/10.1016/s0168-1699(00)00139-3
36. Zhao, S., Zhang, T., Ma, S., Chen, M. (2022). Dandelion Optimizer: A nature-inspired metaheuristic algorithm for engineering applications. Engineering Applications of Artificial Intelligence, 114, 105075. https://doi.org/10.1016/j.engappai.2022.105075
37. Paliwal, N., Srivastava, L., Pandit, M. (2020). Application of grey wolf optimization algorithm for load frequency control in multi-source single area power system. Evolutionary Intelligence, 15 (1), 563–584. https://doi.org/10.1007/s12065-020-00530-5
38. Dorigo, M., Blum, C. (2005). Ant colony optimization theory: A survey. Theoretical Computer Science, 344 (2-3), 243–278. https://doi.org/10.1016/j.tcs.2005.05.020
39. Poli, R., Kennedy, J., Blackwell, T. (2007). Particle swarm optimization. Swarm Intelligence, 1 (1), 33–57. https://doi.org/10.1007/s11721-007-0002-0
40. Bansal, J. C., Sharma, H., Jadon, S. S., Clerc, M. (2014). Spider Monkey Optimization algorithm for numerical optimization. Memetic Computing, 6 (1), 31–47. https://doi.org/10.1007/s12293-013-0128-0

41. Yeromina, N., Kurban, V., Mykus, S., Peredrii, O., Voloshchenko, O., Kosenko, V. et al. (2021). The Creation of the Database for Mobile Robots Navigation under the Conditions of Flexible Change of Flight Assignment. International Journal of Emerging Technology and Advanced Engineering, 11 (5), 37–44. https://doi.org/10.46338/ijetae0521_05
42. Maccarone, A. D., Brzorad, J. N., Stone, H. M. (2008). Characteristics And Energetics of Great Egret and Snowy Egret Foraging Flights. Waterbird, 31 (4), 541–549. https://doi.org/10.1675/1524-4695-31.4.541
43. Ramaji, I. J., Memari, A. M. (2018). Interpretation of structural analytical models from the coordination view in building information models. Automation in Construction, 90, 117–133. https://doi.org/10.1016/j.autcon.2018.02.025
44. Pérez-González, C. J., Colebrook, M., Roda-García, J. L., Rosa-Remedios, C. B. (2019). Developing a data analytics platform to support decision making in emergency and security management. Expert Systems with Applications, 120, 167–184. https://doi.org/10.1016/j.eswa.2018.11.023
45. Chen, H. (2018). Evaluation of Personalized Service Level for Library Information Management Based on Fuzzy Analytic Hierarchy Process. Procedia Computer Science, 131, 952–958. https://doi.org/10.1016/j.procs.2018.04.233
46. Chan, H. K., Sun, X., Chung, S.-H. (2019). When should fuzzy analytic hierarchy process be used instead of analytic hierarchy process? Decision Support Systems, 125, 113114. https://doi.org/10.1016/j.dss.2019.113114
47. Osman, A. M. S. (2019). A novel big data analytics framework for smart cities. Future Generation Computer Systems, 91, 620–633. https://doi.org/10.1016/j.future.2018.06.046
48. Nechyporuk, O., Sova, O., Shyshatskyi, A., Kravchenko, S., Nalapko, O., Shknai, O. et al. (2023). Development of a method of complex analysis and multidimensional forecasting of the state of intelligence objects. Eastern-European Journal of Enterprise Technologies, 2 (4 (122)), 31–41. https://doi.org/10.15587/1729-4061.2023.276168
49. Merrikh-Bayat, F. (2015). The runner-root algorithm: A metaheuristic for solving unimodal and multimodal optimization problems inspired by runners and roots of plants in nature. Applied Soft Computing, 33, 292–303. https://doi.org/10.1016/j.asoc.2015.04.048
50. Poliarush, O., Krepych, S., Spivak, I. (2023). Hybrid approach for data filtering and machine learning inside content management system. Advanced Information Systems, 7 (4), 70–74. https://doi.org/10.20998/2522-9052.2023.4.09
51. Balochian, S., Baloochian, H. (2019). Social mimic optimization algorithm and engineering applications. Expert Systems with Applications, 134, 178–191. https://doi.org/10.1016/j.eswa.2019.05.035
52. Lenord Melvix, J. S. M. (2014). Greedy Politics Optimization: Metaheuristic inspired by political strategies adopted during state assembly elections. 2014 IEEE International Advance Computing Conference (IACC), 1157–1162. https://doi.org/10.1109/iadcc.2014.6779490

53. Moosavian, N., Roodsari, B. K. (2014). Soccer League Competition Algorithm, a New Method for Solving Systems of Nonlinear Equations. International Journal of Intelligence Science, 4 (1), 7–16. https://doi.org/10.4236/ijis.2014.41002
54. Hayyolalam, V., Pourhaji Kazem, A. A. (2020). Black Widow Optimization Algorithm: A novel meta-heuristic approach for solving engineering optimization problems. Engineering Applications of Artificial Intelligence, 87, 103249. https://doi.org/10.1016/j.engappai.2019.103249
55. Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-qaness, M. A. A., Gandomi, A. H. (2021). Aquila Optimizer: A novel meta-heuristic optimization algorithm. Computers & Industrial Engineering, 157, 107250. https://doi.org/10.1016/j.cie.2021.107250
56. Hodlevskyi, M., Burlakov, G. (2023). Information technology of quality improvement planning of process subsets of the spice model. Advanced Information Systems, 7 (4), 52–59. https://doi.org/10.20998/2522-9052.2023.4.06
57. Askari, Q., Younas, I., Saeed, M. (2020). Political Optimizer: A novel socio-inspired meta-heuristic for global optimization. Knowledge-Based Systems, 195, 105709. https://doi.org/10.1016/j.knosys.2020.105709
58. Mohamed, A. W., Hadi, A. A., Mohamed, A. K. (2019). Gaining-sharing knowledge based algorithm for solving optimization problems: a novel nature-inspired algorithm. International Journal of Machine Learning and Cybernetics, 11 (7), 1501–1529. https://doi.org/10.1007/s13042-019-01053-x
59. Gödri, I., Kardos, C., Pfeiffer, A., Váncza, J. (2019). Data analytics-based decision support workflow for high-mix low-volume production systems. CIRP Annals, 68 (1), 471–474. https://doi.org/10.1016/j.cirp.2019.04.001
60. Harding, J. L. (2013). Data quality in the integration and analysis of data from multiple sources: some research challenges. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XL-2/W1, 59–63. https://doi.org/10.5194/isprsarchives-xl-2-w1-59-2013
61. Orouskhani, M., Orouskhani, Y., Mansouri, M., Teshnehlab, M. (2013). A Novel Cat Swarm Optimization Algorithm for Unconstrained Optimization Problems. International Journal of Information Technology and Computer Science, 5 (11), 32–41. https://doi.org/10.5815/ijitcs.2013.11.04
62. Karaboga, D., Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of Global Optimization, 39 (3), 459–471. https://doi.org/10.1007/s10898-007-9149-x
63. Fister, I., Fister, I., Yang, X.-S., Brest, J. (2013). A comprehensive review of firefly algorithms. Swarm and Evolutionary Computation, 13, 34–46. https://doi.org/10.1016/j.swevo.2013.06.001
64. Sova, O., Radzivilov, H., Shyshatskyi, A., Shvets, P., Tkachenko, V., Nevhad, S. et al. (2022). Development of a method to improve the reliability of assessing the condition of the monitoring object in special-purpose information systems. Eastern-European Journal of Enterprise Technologies, 2 (3 (116)), 6–14. https://doi.org/10.15587/1729-4061.2022.254122

65. Khudov, H., Khizhnyak, I., Glukhov, S., Shamrai, N., Pavlii, V. (2024). The method for objects detection on satellite imagery based on the firefly algorithm. Advanced Information Systems, 8 (1), 5–11. https://doi.org/10.20998/2522-9052.2024.1.01
66. Owaid, S. R., Zhuravskyi, Y., Lytvynenko, O., Veretnov, A., Sokolovskyi, D., Plekhova, G. et al. (2024). Development of a method of increasing the efficiency of decision-making in organizational and technical systems. Eastern-European Journal of Enterprise Technologies, 1 (4 (127)), 14–22. https://doi.org/10.15587/1729-4061.2024.298568
67. Tyurin, V., Bieliakov, R., Odarushchenko, E., Yashchenok, V., Shaposhnikova, O., Lyashenko, A. et al. (2023). Development of a solution search method using an improved locust swarm algorithm. Eastern-European Journal of Enterprise Technologies, 5 (4 (125)), 25–33. https://doi.org/10.15587/1729-4061.2023.287316
68. Yakymiak, S., Vdovytskyi, Y., Artabaiev, Y., Degtyareva, L., Vakulenko, Y., Nevhad, S. et al. (2023). Development of the solution search method using the population algorithm of global search optimization. Eastern-European Journal of Enterprise Technologies, 3 (4 (123)), 39–46. https://doi.org/10.15587/1729-4061.2023.281007
69. Mohammed, B. A., Zhuk, O., Vozniak, R., Borysov, I., Petrozhalko, V., Davydov, I. et al. (2023). Improvement of the solution search method based on the cuckoo algorithm. Eastern-European Journal of Enterprise Technologies, 2 (4 (122)), 23–30. https://doi.org/10.15587/1729-4061.2023.277608
70. Raskin, L., Sukhomlyn, L., Sokolov, D., Vlasenko, V. (2023). Multi-criteria evaluation of the multifactor stochastic systems effectiveness. Advanced Information Systems, 7 (2), 63–67. https://doi.org/10.20998/2522-9052.2023.2.09
71. Arora, S., Singh, S. (2018). Butterfly optimization algorithm: a novel approach for global optimization. Soft Computing, 23 (3), 715–734. https://doi.org/10.1007/s00500-018-3102-4
72. Mamoori, G. A., Sova, O., Zhuk, O., Repilo, I., Melnyk, B., Sus, S. et al. (2023). The development of solution search method using improved jumping frog algorithm. Eastern-European Journal of Enterprise Technologies, 4 (3 (124)), 45–53. https://doi.org/10.15587/1729-4061.2023.285292
73. Alieinykov, I., Thamer, K. A., Zhuravskyi, Y., Sova, O., Smirnova, N., Zhyvotovskyi, R. et al. (2019). Development of a method of fuzzy evaluation of information and analytical support of strategic management. Eastern-European Journal of Enterprise Technologies, 6 (2 (102)), 16–27. https://doi.org/10.15587/1729-4061.2019.184394
74. Shyshatskyi, A. V., Zhuk, O. V., Neronov, S. M., Protas, N. M., Kashkevych, S. O. (2024). Sukupnist metodyk pidvyshchennia operatyvnosti pryiniattia rishen z vykorystanniam metaevrystychnykh alhorytmiv. Modern Aspects of Science. Scientific Perspectives, 529–557. Available at: http://perspectives.pp.ua/public/site/mono/mono-40.pdf
75. Shyshatskyi, A. V., Matsyi, O. B., Yashchenok, V. Zh., Trotsko, O. O., Kashkevych, S. O. (2024). Sukupnist metodyk pidvyshchennia operatyvnosti pryiniattia rishen z vykorystanniam

kombinovanykh metaevrystychnykh alhorytmiv. Modern Aspects of Science. Scientific Perspectives, 558–594. Available at: http://perspectives.pp.ua/public/site/mono/mono-40.pdf

76. Kashkevich, S., Dmytriieva, O., Trotsko, O., Shknai, O., Shyshatskyi, A. (2023). Mathematical model of information conflict of information networks. ScienceRise, 1, 3–13. https://doi.org/10.21303/2313-8416.2024.003395
77. Kashkevych, S. O., Dmytriieva, O. I., Yefymenko, O. V., Pliekhova, H. A., Shyshatskyi, A. V. (2024). Metody otsinky stanu skladnykh dynamichnykh obiektivz vykorystanniam bioinspirovanykh alhorytmiv. Modern Aspects of Science. Scientific Perspectives, 138–177. Available at: http://perspectives.pp.ua/public/site/mono/mono-40.pdf
78. Kashkevych, S. O., Dmytriieva, O. I., Trotsko, O. O., Shknai, O. V., Shyshatskyi, A. V. (2024). Metod samoorhanizatsii informatsiinykh merezh v umovakh destabilizuiuchykh vplyviv. The development of technical, agricultural and applied sciences as the main factor in improving life. Boston: Primedia eLaunch, 192–218. https://doi.org/10.46299/ISG.2024.MONO.TECH.2.7.2
79. Shyshatskyi, A., Dmytriieva, O., Lytvynenko, O., Borysov, I., Vakulenko, Y., Mukashev, T. et al. (2024). Development of a method for assessing the state of dynamic objects using a combined swarm algorithm. Eastern-European Journal of Enterprise Technologies, 3 (4 (129)), 44–54. https://doi.org/10.15587/1729-4061.2024.304131
80. Kashkevich, S., Litvinenko, O., Shyshatskyi, A., Salnyk, S., Velychko, V. (2024). The method of self-organization of information networks in the conditions of the complex influence of destabilizing factors. Advanced Information Systems, 8 (3), 59–71. https://doi.org/10.20998/2522-9052.2024.3.07
81. Kashkevych, S. O., Dmytriieva, O. I., Plekhova, H. A., Protas, N. M., Neronov, S. M., Shyshatskyi, A. V. (2024). Naukovo-metodychnyi pidkhid z pidvyshchennia operatyvnosti obrobky riznotypnykh danykh z vykorystanniam metaevrystychnykh alhorytmiv. Modern Aspects of Science. Scientific Perspectives, 510–543. Available at: http://perspectives.pp.ua/public/site/mono/mono-40.pdf
82. Litvinenko, O., Kashkevich, S., Dmytriieva, O., Neronov, S., Plekhova, G.; Shyshatskyi, A. (Ed.) (2024). The method of self-organization of information networks in the conditions of destabilizing influences. Information and control systems: modelling and optimizations. Kharkiv: TECHNOLOGY CENTER PC, 3–34. https://doi.org/10.15587/978-617-8360-04-7.ch1
83. Kashkevich, S., Dmytriieva, O., Neronov, S., Plekhova, G., Zhyvylo, Y.; Shyshatskyi, A. (Ed.) (2024). The development of management methods based on bio-inspired algorithms. Information and control systems: modelling and optimizations. Kharkiv: TECHNOLOGY CENTER PC, 35–69. https://doi.org/10.15587/978-617-8360-04-7.ch2
84. Kashkevich, S., Plekhova, G., Kuchuk, N., Kuvshynov, O., Veretnov, A., Yefymenko, O.; Shyshatskyi, A. (Ed.) (2024). The development of methods for evaluating the state of complex technical systems using artificial intelligence theory. Information and control systems: modelling and optimizations. Kharkiv: TECHNOLOGY CENTER PC, 70–101. https://doi.org/10.15587/978-617-8360-04-7.ch3

85. Kashkevich, S., Plekhova, G., Yefymenko, O., Kuchuk, H., Davydov, V., Beketov, Y.; Shyshatskyi, A. (Ed.) (2024). The development of methods of learning artificial neural networks of intelligent decision-making support systems. Information and control systems: modelling and optimizations. Kharkiv: TECHNOLOGY CENTER PC, 102–137. https://doi.org/10.15587/978-617-8360-04-7.ch4
86. Kashkevich, S., Dmytriiev, I., Shevchenko, I., Lytvynenko, O., Shabanova-Kushnarenko, L., Apenko, N; Shyshatskyi, A. (Ed.) (2024). Scientific-method apparatus for improving the efficiency of information processing using artificial intelligence. Information and control systems: modelling and optimizations. Kharkiv: TECHNOLOGY CENTER PC, 138–167. https://doi.org/10.15587/978-617-8360-04-7.ch5