УДК 004.021
Doi: 10.31772/2712-8970-2021-22-1-32-46

Для цитирования: Карцан И. Н., Ефремова С. В. Мультиверсионная модель программного обеспечения систем управления космическим аппаратом с ранжированием принятия решения // Сибирский аэрокосмический журнал. 2021. Т. 22, № 1. С. 32–46. Doi: 10.31772/2712-8970-2021-22-1-32-46.

For citation: Kartsan I. N., Efremova S. V. Multiversion model of software control systems for space vehicles with range of decision-making // Siberian Aerospace Journal. 2021, Vol. 22, No. 1, P. 32–46. Doi: 10.31772/2712-8970-2021-22-1-32-46.

Мультиверсионная модель программного обеспечения систем управления космическим аппаратом с ранжированием принятия решения*

И. Н. Карцан[1, 2, 3**], С. В. Ефремова[2]

[1]ФГБУН ФИЦ «Морской гидрофизический институт РАН»
Российская Федерация, 299011, г. Севастополь, ул. Капитанская, 2
[2]Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева
Российская Федерация, 660037, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31
[3]Севастопольский государственный университет
Российская Федерация, 299053, г. Севастополь, ул. Университетская, 33
[**]E-mail: kartsan2003@mail.ru

В статье представлена мультиверсионная модель с ранжированием альтернатив в порядке предпочтения с учетом зависимости атрибутов при проектировании программного обеспечения для системы управления космическими аппаратами различного класса. Применяемое программное обеспечение с набором алгоритмов, базирующихся на общей схеме метода ветвей и границ, позволяет определять точное решение оптимизационной задачи.

Для достижения наибольшей надежности программной составляющей систем управления космическими аппаратами, построенной с использованием методологии мультиверсионного программирования, в единую структуру объединяется большое количество версий программных модулей.

В то время как программные комплексы даже без введения избыточных элементов характеризуются как сложные системы, говорить о широком использовании переборных методов для их формирования не приходится.

Использование предложенного модифицированного метода упорядоченного предпочтения через сходство с идеальным решением позволит решить задачу выбора лучшей вычислительной системы из ряда доступных систем. Данный подход становится все более возможным по причине колоссального прогресса в технологиях проектирования и производства вычислительной техники. Даже так называемые персональные компьютеры предоставляют вычислительные возможности, которые некоторое время назад казались невозможными для компьютеров – представителей значительно более мощного класса вычислительной техники – суперкомпьютеров.

Ключевые слова: мультиверсионная модель, программное обеспечение, системы управления космическими аппаратами, алгоритм, многоатрибутивные методы принятия решений.

* Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 19-37-90076.

Работа выполнена в рамках государственного задания по теме № 0555-2021-0005.

The research was carried out with the financial support of the RFBR in the framework of scientific project No. 19-37-90076.

This study was supported by the Russian Federation State Task № 0555-2021-0005.

Multiversion model of software control systems for space vehicles with range of decision-making

I. N. Kartsan[1, 2**], S. V. Efremova[2]

[1]Marine Hydrophysical Institute, Russian Academy of Sciences
2, Kapitanskaya St., Sevastopol, 299011, Russian Federation
[2]Reshetnev Siberian State University of Science and Technology
31, Krasnoiarskii Rabochi Prospekt, Krasnoyarsk, 660037, Russian Federation
[3]Sevastopol State University
33, Universitetskaya St., Sevastopol, 299053, Russian Federation
[**]E-mail: kartsan2003@mail.ru

The paper presents a multi-version model with ranking of alternatives in order of preference, taking into account the dependence of the attributes in the design of software for spacecraft control systems of various classes. The applied software with a set of algorithms, based on the general scheme of the method of branches and borders allow determining the exact solution of the optimization problem.

To achieve the highest reliability of the software component of spacecraft control systems built with the use of multi-version programming methodology, a large number of versions of software modules are combined into a single structure.

While software complexes even without introduction of redundant elements are characterized as complex systems, there is no need to speak about wide use of enumerative methods for their formation.

Using the proposed modified method of ordered preference through similarity to an ideal solution, will allow to solve the problem of choosing the best computing system from a number of available systems. This approach is becoming increasingly possible because of the tremendous progress in computing design and manufacturing technology. Even the so-called personal computers provide computational capabilities that some time ago seemed impossible even for computers representing a much more powerful class of computing equipment - supercomputers.

Keywords: multi-version model, software, spacecraft control systems, algorithm, multi-attribute decision-making methods.

Introduction. At the stage of designing the appearance of spacecraft control systems, decisions are made on the choice of the composition of the multi-version software of the system using the fuzzy programming method. This allows the designer to set the degree of "attribute preference" and the possible "percentage of attainability" of goals when choosing one or another option for forming the composition of multi-version software [1–3]. The result, as a rule, is a set of non-dominated solutions to the problem of forming the multi-version software of the control system.

The problem of forming a complex of multi-version software for spacecraft control systems is an objective of conditional optimization of a monotonic pseudo-Boolean function.

The most universal of all discrete optimization methods is the exhaustive search method when all feasible solutions are examined to find the best value of the objective function [4].

Algorithms of two main types have been developed in this direction: implicit exhaustive search, where an increase in the speed of finding the optimal solution is realized through narrowing the search area and reducing the total number of calculations in the algorithm and a branch-and-bound method scheme with dividing the search area into smaller sub domains.

Among the exact algorithms of the first type one should single out procedures that use to enhance the computational capabilities of exact algorithms, the idea of narrowing the search area due to the features of the functions-elements of the optimization problem being solved and the fastest traversal of the search area by neighboring points using relations on the space of Boolean variables [5].

The algorithms of the branch-and-bound scheme [6] include algorithms for finding solutions on sub cubes [7–8]. Procedures of this type differ only in the way of organizing the division of the solution search area into sub cubes that is in the way of representing the original optimization problem in the form of a certain number of tasks of lower dimension [9].

A large number of modules of the multi-version software of spacecraft control systems their redundant versions as well as limitations of real needs such as the cost of development, implementation and modification of the system pose the designer the task of making decisions on the choice of the optimal composition of the multi-version software of the system taking into account a number of attributes [10–13].

Due to the existence of a sufficiently large number of alternatives, their assessment is largely influenced by the qualities of the decision-maker himself and his subjective preferences. Therefore when making a choice of the best alternative from a number of proposed ones it is necessary to take into account the subjectivity in the assessments of the decision-maker. [14].

General ranking model when choosing the optimal solution. In new information technologies decision-making is considered to be a set of decisions in conditions of certainty which make it possible to choose unambiguous, consistent, correct decisions based on formalized models of control objects and their environment. The formation of certain probabilistic decisions in conditions of uncertainty should also be considered as decision-making. [11].

The problems of decision support in new information technologies cover all the problems including the class of tasks under the conditions of uncertainty, the final solution of which is carried out outside the used technology. In these cases the information is converted to a form that simplifies and facilitates decision-making by other methods. It can be stated that currently various methods and approaches are used to support decision-making, which together complement each other. Decision making involves choosing a sequence of actions and implementing it. Decision support is based on obtaining multivariate decisions using different methods. The decision support methodology includes a variety of schemes and technologies; it can be implemented partially or completely using software and information systems. The simplest decision-making model includes four main cyclically repeating stages:

- collection, analysis and transformation of data;
- obtaining options for solutions (alternatives);
- development of criteria for evaluating decisions;
- selection of one of the options based on the selected criteria.

There are different conditions and situations in which you need to make decisions. There are various levels of decision making. The top one is the conceptual level. It allows you to draw up a generalized diagram. Decision making technology generally includes the following stages:

- emergence of a problem or task requiring a solution;
- formation of the problem at the verbal level;
- search for information necessary for making a decision;
- formalization of task setting; - analysis and processing of information;
- formation of a set of alternatives;
- obtaining forecast estimates;
- evaluation of the results of decision-making.

A task that requires a solution happens at the first stage. As a rule it is formed at the verbal (non-formalized) level of communication. Thereafter a search and collection of information is carried out in an aspect of the given problem or task. At the same time they collect not only the data necessary for the solution but also information about the methods for solving such problems. After that they formalize or formulate the problem at a formal level. [15].

At the stage of information analysis and processing, complex processing is carried out, including computer methods and expert analysis; direct calculation and heuristic methods; optimization methods.

A possible change in conditions is provided for by obtaining sets of solutions. Mutually exclusive solutions are called alternatives. Therefore, for completeness and taking into account changing conditions as a result of the analysis, the formation of alternatives is carried out, which is presented as an important stage in the decision-making scheme. The presentation of the generalized data is carried out in a form that is convenient for making decisions.

The next stage and one of the most important is the stage of obtaining forecast estimates. At this stage, using the available solutions, information about the conditions and methods of obtaining a forecast, forecast estimates are obtained, and the choice of the forecasting method and the forecast verification method are justified.

After receiving a set of data: alternatives, information about the dynamics of conditions, forecast estimates, assessment of forecast reliability, etc., complex processing is performed using an expert approach. This scheme can work with different technologies and control levels.

The connection between the listed stages should be emphasized. They form a hierarchical sequence. If at one of the stages there is no possibility of its implementation, then the transition to the next is not carried out. Decision support means that the group of these methods is aimed not only at obtaining decisions, but also at providing recommendations for the decision maker. Thus, decision support includes three groups of tasks:

1. Getting a set of solutions but not one.
2. Preparation of criteria for evaluating the solutions obtained.
3. Choosing a solution from the available population.

There is a problem of correct extracting the necessary information when making decisions and evaluating them.

The use of modern technologies and decision-making methods requires the use of formalized data. Therefore the effectiveness of methods for obtaining solutions and decision support depends on the formalization of the problem. So when a complex problem is decomposed to an operational level and the conditions for solving a problem at this level are fully formalized, it is effective to use operations research methods to obtain solutions. If decomposition and complete formalization are impossible, methods of statistical evaluation, the theory of fuzzy sets, etc. are used.

The choice of the best option for the formation of the multi-version software of the system from the entire set of possible implementations can be made using the compensation model of multi-attribute decision making, which allows you to perform a general ranking of alternatives based on the order of their preference for individual attributes and the relationship between them and to determine the best option for the formation of multi-version software.

The developed model of decision-making on the composition of the multiversion software of the system assumes the ranking of possible options for their formation in order of preference. The first ranked option is the best. Using only order of preference as input avoids scaling of quality type attributes [16, 17].

This model is based on the ranking of alternatives for individual attributes. On the basis of this "private" ranking the general order of preference for alternatives is determined taking into account all the attributes and the relationship between them.

There is a linear assignment method that allows performing a general ranking of alternatives. [18]. According to this method the total rank was calculated as the sum of the ranks for individual attributes. In this case information about the relationship between attributes was ignored:

$$r_{i_{общ}} = \sum_{j=1}^{n} r_{ij}, i = \overline{1,m}, \tag{1}$$

where m is a number of alternatives; n is a number of attributes; r_{ij} is a rank of i alternative on j attribute; r is a number of ranks ($r = m$).

However for most decision-making tasks and in particular for choosing the option for forming the multi-version software of the system it is important to take into account this dependence [19]. In this regard on the basis of this method a multi-attribute compensation model was developed for the general ranking of alternatives in order of preference taking into account the dependence of the attributes. The idea of compensation in this case is to take into account the dependence between attributes: a change in the value of one of them leads to a change in the values of any other attributes. Let us define the matrix π as a square non-negative matrix m x m, whose elements π_{ik} represent the number (or frequency) of the ranking of the alternative A_i by the r th rank. The matrix π is formed on the basis of the ranking matrix of alternatives for individual attributes D:

$$\pi_{ik} = \sum_{l=1}^{n} I(D_{jl}^{i}) \cdot w_l; i = \overline{1,m}; j = \overline{1,r}, \tag{2}$$

$$I(D_{jl}^{i}) = \begin{cases} 1, & \text{если } D_{jl}^{i} = i, \\ 0, & \text{если } D_{jl}^{i} \neq i, \end{cases} \tag{3}$$

where $I(x)$ is an indicator function; w_l is a weight coefficient of the l st attribute.

For different weights, the elements of the matrix π represent the sum of the weights of the attributes of the corresponding rank. It is assumed that the weights are normalized.

Obviously π_{ik} determines the contribution of the alternative A_i to the overall ranking. The larger the value of $\pi_{ik\ is}$, the more correct is the assignment of the rth rank to the alternative A_i

Let's define the permutation matrix Q as the square matrix m x m, which elements Q_{ir} = 1, if the alternative A_i is assigned a general rank r, and Q_{ir} = 0 otherwise. The objective function can be written as follows:

$$\max_{Qr} \sum_{i=1}^{m} \sum_{j=1}^{r} \pi_{ik} Q_{ir}, \tag{4}$$

under conditions

$$\sum_{j=1}^{r} Q_{ir} = 1; i = \overline{1,m}, \tag{5}$$

$$\sum_{i=1}^{m} Q_{ir} = 1; \; j = \overline{1,r}. \tag{6}$$

Conditions mean that only one rank can be assigned to alternative A_i and rank r can only be assigned to one alternative.

Let us denote the optimal permutation matrix representing the solution of the above linear programming problem as Q *. Then the optimal ordering can be achieved by multiplying the matrix A containing the numbers of alternatives by Q *.

Let's consider an example of a model application of the proposed compensatory multi-attributive general ranking model when choosing one of the three alternatives. Let all attributes have increasing preference that is the higher the value of the attribute is the more preferable the alternative is.

Suppose that the ranks of these alternatives for three separate attributes correspond to those given in Table. 1. So the first alternative has the first rank according to the first attribute, the first rank according to the second, and the second according to the third.

Table 1

Ranking alternatives by individual attributes

attribute		1	2	3
rank	1	A1	A1	A2
	2	A2	A3	A1
	3	A3	A2	A3

Ranking by individual attributes can be represented as a matrix D, the elements of which are the indices of the ranked alternatives:

$$D = \begin{bmatrix} 1 & 1 & 2 \\ 2 & 3 & 1 \\ 3 & 2 & 3 \end{bmatrix}. \tag{7}$$

Based on this matrix one can obtain a matrix π, its elements represent the number of assignments to an alternative of each of the ranks. So the first alternative was assigned the first rank twice, the second rank - once, and the third - never, which corresponds to the value in the first row of the matrix:

$$\pi = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix}. \tag{8}$$

For weighting factors w1 = 0.2, w2 = 0.4, w3 = 0.4 the elements of the matrix π will change as follows:

$$\pi = \begin{bmatrix} 0.2+0.4 & 0.4 & 0 \\ 0.4 & 0.2 & 0.4 \\ 0 & 0.4 & 0.2+0.4 \end{bmatrix} = \begin{bmatrix} 0.6 & 0.4 & 0 \\ 0.4 & 0.2 & 0.4 \\ 0 & 0.4 & 0.6 \end{bmatrix}. \tag{9}$$

The optimal permutation matrix Q * which determines the overall rank of each of the alternatives has the form:

$$Q^* = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (10)$$

It can be seen that the first alternative (the first column corresponds to it) has a general rank equal to one (the first row), the second alternative received the second rank and the third has got the third. Based on the Q * matrix, we get the following order of preferences:

$$A1 \succ A2 \succ A3. \quad (11)$$

The multi-attribute decision making compensation model allows for a general ranking of alternatives in order of preference based on ranking for individual attributes and the relationship between attributes. The merits of this model are its unique advantages in practical application. An expert is required to set the order of preference for alternatives by attributes when collecting data. This approach avoids the difficulties that arise when creating a scale for assessing attributes.

Optimization algorithms.

Implicit brute force algorithms. In the model with sequential organization of software modules, the multi-version software complex is defined as consisting of a set of sequential software modules, for which a set I, *card* (I) =*I* is introduced. Many modules are divided into classes that means many classes of software modules (J, *card* (J) =*J*) are introduced. By referring a certain software module to a certain class, it is assigned a solution to the corresponding intermediate "typical" problem of control or information processing. Combining "standard" software modules into complexes contributes to the formation of the process of solving the general target problem of the control system. In order to implement modules of one class a software module is assigned which in turn to ensure the reliability of execution is implemented using the multi-version programming methodology. Functionally equivalent versions S_j, $(j = \overline{1,J})$ are developed for each software module in accordance with the original specifications. Thus, a vector **S**= $\{S_j\}, (j = \overline{1,J})$, is introduced, its elements are numbers equal to the number of versions of software modules (S_j is the number of versions of a module that solves the problem of class *j* - that implements a module of class *j*).

As a result, the set of all software modules **K** is determined, then *K*=*card* (K) characterizes the total number of software modules included in the designed software package.

In addition the set **B**, *card* (B) =*I* of belonging of tasks to classes is determined, the cardinality of which is equal to the number of tasks in the system and each element of this set is equal to the number of the class to which the task belongs. Thus, the element B_i of the set **B** represents the number of a typical module; with its help i-th control problem in the general control complex is solved.

We are introducing Boolean variables:

$$X_s^i = \begin{cases} 1, \textit{ in order to realise } i-th\ (i = \overline{1,I})\ \textit{task} \\ \textit{we use } s-th\ (s = \overline{1,S_{B_i}})\ \textit{version of module } B_i \\ 0, \textit{ in opposite case.} \end{cases} \quad (12)$$

The introduced variables unfold into a vector of participation, which formally describes the possible variants of the formation of the composition of the multi-version complex of programs. The task of the optimal formation of multi-version software is to find a set of multi-versions, which determines the greatest reliability of the software package, subject to a certain level of financial costs for creating the system [7; 8; 20]. The problem formulated in this way using the previously introduced notation was formalized as follows:

$$max R_{NVS1}(X) = \prod_{i=1}^{l} R_i(X), \quad (13)$$

where $R_i(X) = 1 - \prod_{s=1}^{S_{B_i}} (1 - R_{B_i S})^{X_S^i}$ is assessment of the reliability of the i-th software module as part of a multi-version software package, $R_{B_i S}$ is an assessment of the reliability of the s-th version of the i-th

software module. The limitation on the cost of the designed software package is

$$C_{NVS1}(X) \le B, \tag{14}$$

where

$$C_{NVS1}(X) = \sum_{i=1}^{I} \sum_{s=1}^{S_{B_i}} X_S^i \cdot C_{B_i S}, \tag{15}$$

$C_{B_i S}$ is the level of financial costs for the implementation of the s-th version of the i-th software module of the designed software package.

Thus in the formal goal setting of this type, the problem of the optimal formation of multi-version software is formulated as finding the values of the components of the vector X so that the reliability function of the software package (13) takes its greatest value if the cost function (14) does not exceed a certain value B.

For a single-valued description of the problems of forming multi-version software complexes as well as for converting two indices of Boolean variables defined in (12) into one number of the participation vector component, it is necessary to determine the procedure for forming the participation vector and accordingly an algorithm for determining the indices of the components of this vector [21–23].

In the second optimization model, a model with sequential-parallel organization of program modules, a multi-version software package is also considered to consist of a set of control problems of sequential execution (multitude **I**, *card*(I)=*I*). Like the first model software modules are divided into *J* types according to the required functions, that is a set of typical control problems is determined or a set of classes of problems – **J**, *card*(**J**)=*J*.

But unlike the model with a sequential organization of modules, each task $i=\overline{1, I}$ of the software package is implemented not by one module, but by a certain set of software modules which is specified in the vector $\mathbf{J}_i$, где $\mathbf{J}_i = \left\{ j_1^i, ..., j_{J_i}^i \right\}$ therefore *card* ($\mathbf{J}_i$) = Ji is the number of software modules involved in solving the i-th problem and $j_k^i, k = \overline{1, J_i}$ is the number of the class to which the k-th program module belongs.

Each typical task from the set **J** is implemented using a fault-tolerant software module developed using the multi-version programming approach, that is, each element of the set J is associated with a set of versions of the module it defines ($\mathbf{V}_k$ is a multitude of versions *k*, $k=\overline{1, J}$, S_k=*card*($\mathbf{V}_k$) is a number of module versions *k*). Thus for each software module in accordance with the original specifications a S_j, ($j=\overline{1, J}$) of functionally equivalent versions is developed, that means vector **S** = {S_j}, ($j=\overline{1, J}$) is introduced. Its elements are numbers equal to the number of versions of software modules (S_j is a number of module versions realizing *j* class module).

As in the first model for a formal description of the structure of the designed complex of multi-version software, a vector of participation *X* is introduced, the components of which are such Boolean variables X_{kS}^i as

$$X_{ks}^i = \begin{cases} 1, \textit{ in order to realise } i-th\,(i=\overline{1, I})\textit{ task} \\ \textit{we use } s-th\,(s=\overline{1, S_{J_k^l}})\textit{ version of module } k=(k=\overline{1, J_l}), \\ 0, \textit{ in opposite case.} \end{cases} \tag{16}$$

Similarly to the model with sequential organization of software modules the task of optimal formation of multi-version software in this model is to determine the set of multi-versions that characterize the greatest reliability of the software package without exceeding the established budget for creating the system, that is, the optimization problem is formally posed as follows:

$$max R_{NVS2}(X) = \prod_{i=1}^{l} R_i(X), \tag{17}$$

where

$$R_i(X) = \prod_{k=1}^{J_i} R_k^i(X), \tag{18}$$

and

$$R_k^i(X) = 1 - \prod_{s=1}^{S_{j_k^i}} (1 - R_{j_k^i,s})^{X_{ks}^i}, \tag{19}$$

whereas $R_i(X)$ is assessment of the reliability of the i-th set of software modules as part of a multi-version software package, $R_k^i(X)$ is reliability assessment of the k-th software module as part of the i-th set, $R_{j_k^i,s}$ is assessment of the reliability of the s-th version of the k-th software module as part of the i-th set.

The limitation on the cost of the designed software package is determined by the formula

$$C_{NVS2}(X) \le B\ , \tag{20}$$

where

$$C_{NVS2}(X) = \sum_{i=1}^{I} \sum_{k=1}^{J_i} \sum_{s=1}^{S_{j_k^i}} X_{kS}^i \cdot C_{j_k^i,s}, \tag{21}$$

$C_{j_k^i,s}$ is the level of costs for the implementation of the s-th version of the k-th software module as part of the i-th set of the designed multi-version software complex. Obviously if among the set of all boundary points we select the two closest to the point X_o all coordinates of which are equal to zero and the boundary point farthest from X_o, and determine the levels I_{min} and I_{max} of the point X_o corresponding to these two points then it becomes possible to narrow the region search for solutions since undoubtedly the solution of the obtained optimization problem will belong to the set defined by the following formula

$$S = \bigcup_{i=I_{\min}}^{I_{\max}} O_i(X^o), \tag{22}$$

where X^o is a point in the space of Boolean variables such as that $X_i^o = 0, i = \overline{1,n}$.

Thus, to find a solution to the problem it is enough to determine the levels I_{min} and I_{max} of the point X_o and compare the values of the objective function in the elements of the set *S*, defined by the formula (22).

This property makes it possible to significantly reduce the computational costs of finding an exact solution thereby speeding up the execution of the search procedure. The cardinality of a set *S* is determined by the following formula

$$cardS = \sum_{k=I_{\min}}^{I_{\max}} C_n^k, \tag{23}$$

where C_n^k is the number of combinations of n elements by k.

It is also indisputable that due to the additive nature of functions (15) and (19) the boundary point closest to the point X_o will be characterized by the correspondence of its unit coordinates to the largest summands of these expressions. That is in order to reach the boundary point nearest to it from X_o it is necessary to follow the path of the greatest increase of functions (15) and (21).

Algorithm 1. Algorithm for determining the I_{min} level of a point X_o.

1. We take up I =0, $X \in B_2^n : X_i = 0, i = \overline{1,n}.$

2. We form a set of values of versions of software modules of the designed system *C*, card (C)=n, where n is defined as $n = \sum_{i=1}^{I} S_{B_i}$ or $n_{NVS2} = \sum_{i=1}^{I} (\sum_{k=1}^{M_i} S_{j_k^i})$, depending on the problem being solved.

3. From the hypothesis $C_k = maxC_i, i = \overline{1,n},\ X_i = 0$ we determine k.

4. We define $X_k = 1$.

5. If X_k belongs to the set of feasible solutions, then we set I=I+1 and go to 3step.

The algorithm for determining the boundary point farthest from X_o (the algorithm for determining the level I_{max} of point X_o) is constructed similarly to algorithm 1, except that in step 3, k should be determined from the condition of the minimum increase in the function that k is determined so

that $C_k = minC_i, i = \overline{1,n}, X_i = 0$.

The search scheme on the set of solutions defined by expression (22) is implemented in the truncated exhaustive search algorithm.

Algorithm 2. Truncated brute force algorithm.

1. We define I_{min} и I_{max}, corresponding to the problem being solved.
2. We take up $I = I_{min}$.
3. We define such X_I^* vector as $R(X_I^*) = maxR(X),\ X \in O_I(X^0)$.
4. We repeat points 2, 3 for every $I = \overline{I_{min}, I_{max}}$.
5. We take vector X^* as a solution of the task, due to he condition $R(X^*) = maxR(X_I^*),\ I = \overline{I_{min}, I_{max}}$.

It is also worth highlighting the following property of pseudo-Boolean functions (15) and (21): if two adjacent to each other points X and Y differ in the value of some i-th component, whereas X_i=0, то $C(Y)=C(X)+ C_i$,where C_i is the cost of including the multi-version in the structure corresponding to the i-th component of the participation vector. Thus the form of the constraint in the constructed optimization problems allows one to get rid of the complete calculation of expressions (15) and (21) at each point, if the traversal of the solution search area is organized as moving along neighboring points. The entire value of the constraint function for each of the models is calculated at some starting point of the search. Nevertheless the subsequent values of this function are obtained by adding or subtracting the corresponding value when going through neighboring points.

In order to utilize the above-described property of the constructed optimization problem, a procedure for traversing the graph of Boolean variables by neighboring points was implemented, and each point of the solution domain is viewed by the traversal algorithm only once, which avoids increasing the computational complexity of search procedures. This algorithm for traversing the search area is based on the ability to represent any subcube as a union of its lower base point and a union of disjoint subcubes of lower dimensions. This way of traversing the search area is implemented in an implicit search algorithm with traversing the search area by neighboring points.

Algorithm 3. Algorithm of truncated implicit enumeration with traversal of the search area by neighboring points.

1. We define parameters I_{min} and I_{max}, corresponding to the problem being solved.
2. We take up $X \in B_2^n : X_i = 1 \forall i \le I_{min}, k = 0$.
3. Using an implicit enumeration algorithm with traversing the search area by neighboring points, we determine the point X_k^* that gives the largest value of the objective function at this stage.
4. We generate the main point $X \in O_{X^0}(I_{min}), k = k + 1$.
5. We repeat steps $C_n^{I_{min}}$ times, where C_n^k is a number of from n up to k.
6. We take the point $X^* = maxX_k^*, k = \overline{1, C_n^{I_{min}}}$. for a task solution

Branch and Bound Scheme Algorithms. In the second group of exact search procedures, the initial problem is divided into several sub problems of smaller dimension by splitting the search area B_2^n . The partitioning of the solution search area, in turn, is implemented as a partition of the space of Boolean variables into a set of disjoint subcubes that completely cover the entire space. The general scheme of algorithms built using this method is as follows.

Algorithm 4. **Branch and bound method diagram (splitting a Boolean hypercube into subcubes).**

1. The search area represented (in the general case) by a sub cube in a binary space, is divided into *R* disjoint sub cubes.
2. The belonging of the boundary points to each of the sub cubes is determined.
3. If at least one boundary point belongs to the sub cube r-th $(r = \overline{1, R})$, then using the methods of implicit exhaustive search, the sub cube point X_r is found and stored that gives the best value of the objective function that satisfies the constraints.
4. The best solution chosen from local ones is taken $f(X^*) = min, r = \overline{1, R}$.

In the general case, two methods are proposed for dividing a zero hypercube into sub cubes: a partition of B_2^n into $2^{n-n_{sub}}$ sub cubes of the same dimension $n_{sub} < n$ and a method for recursively dividing sub cubes into two sub cubes equal in cardinality, the dimension of which is obviously one less than the dimension of the sub cube being divided. The partition of the space of Boolean variables into sub cubes is based on the corresponding property of the vector-mask.

Conclusion. Optimization tasks, to which the task of forming a complex of multi-version software is reduced, are tasks of the so-called "knapsack" type. However the peculiarities of the objective functions in the optimization models do not allow for their efficient implementation to use the algorithms for solving the knapsack problem developed earlier.

Utilization of the properties of the space of Boolean variables, including the properties and ratios of sub cubes, makes it possible to develop effective regular procedures for the implementation of the constructed optimization models that determine the exact solution of optimization problems.

The choice of the best variant of the formation of the multi-version software of the system is based on the implementation of multivariate decisions when using various methods of multi-attributive decision-making. The multi-attribute decision making compensation model takes into account information about the relationship between attributes. The idea of compensation in this case is to take into account the dependence between attributes: a change in the value of one of them leads to a change in the values of any other attributes. The modified multi-attributive method of ordered preference through similarity to the ideal solution allows finding the best option for the formation of multi-version software by systems in a variety of alternatives of any cardinality.

Experimental data show the advantage in time of searching for a solution to the implicit enumeration algorithm with traversing the search area by neighboring points over the algorithms of the branch-and-bound method.

Библиографические ссылки

1. Волков В. А. Многоатрибутивный выбор компонент отказоустойчивого программного обеспечения // Вестник университетского комплекса. 2006. Вып. 8 (22). С. 208–211.

2. Ковалев И. В., Савин С. В. Оптимальное формирование избыточной структуры для отказоустойчивых информационных систем // Исследовано в России. 2004. Т. 7. С. 1123–1129.

3. The hardware and software implementation of the adaptive platform for an onboard spacecraft control system / I. N. Kartsan, A. O. Zhukov, O. A. Platonov, S. V. Efremova // Journal of Physics : Conference Series. 2019. Vol. 1399, No. 3. P. 033071.

4. Choice of optimal multiversion software for a small satellite ground-based control and command complex / I. N. Kartsan, S. V. Efremova, V. V. Khrapunova, M. I. Tolstopiatov // IOP Conference Series: Materials Science and Engineering. 2018. Vol. 450, No. 2. P. 022015.

5. Formation of optimal composition of the modules of single-function multiversion software for automated control system of the satellite communication system / V. I. Kudymov, V. V. Brezitskaya, P. V. Zelenkov, I. N. Kartsan, Yu. N. Malanina // IOP Conference Series: Materials Science and Engineering. 2018. Vol. 450, No. 5. P. 052009.

6. Kovalev I., Davydenko O. Interactive system for spacecraft technological control cycle construction // Program and Abstracts of Int. Symposium SOR'96. TU-Braunschweig (4–6 Sept. 1996). 1996. P. 195.

7. Stupina A. Realization of conventional pattern of random search methods in the space of Boolean variables // Optimization Days. Montreal, 1997. P. 98–112.

8. Юнусов Р. В. Моделирование программных архитектур автоматизированных систем управления // Управляющие и вычислительные системы. Новые технологии : материалы Всерос. электрон. науч.-техн. конф. Вологда : ВоГТУ, 2001. С. 60–61.

9. Stupina A., Antamoshkin A. The random search algorithms in the space of Boolean variables // Symp. OR'97. Jena, 1997. P. 112–118.

10. How to use neural network and web technologies in modeling complex technical systems / M. G. Semenenko, I. V. Kniazeva, L. S. Beckel et al. // IOP Conference Series : Materials Science and Engineering. 2019. Vol. 537, No. 3. P. 032095.

11. Семенько Т. И. Многоатрибутивный подход к формированию программного обеспечения отказоустойчивых систем управления // Успехи современного естествознания, 2004.
Вып. 6. С. 32–33.

12. Ковалев И. В., Царев Р. Ю. Комбинированный метод формирования мультиверсионного программного обеспечения управления космическими аппаратами // Авиакосмическое приборостроение. 2006. № 9. С. 8–14.

13. Царев Р. Ю. Многоатрибутивное принятие решений в мультиверсионном проектировании : монография. Красноярск : ИПЦ КГТУ, 2005. 156 с.

14. Efremova S. V., Kartsan I. N., Zhukov A. O. An ordered ranking multi-attributive model for decision-making systems with attributes of control systems software // IOP Conference Series: Materials Science and Engineering. 2021. Vol. 1047. P. 012068.

15. Kartsan I. N. Models for Estimating the Reliability of the Software of an Onboard Control System // Research journal of pharmaceutical biological and chemical sciences. 2018. Vol. 9, No. 5. P. 2357–2367.

16. Царев Р. Ю. Анализ качественных и количественных атрибутов на этапе проектирования отказоустойчивых программных систем // Системы управления и информационные технологии. 2006. № 3 (25). С. 95–101.

17. Царев Р. Ю. Компенсационная модель многоатрибутивного принятия решений при формировании информационных систем управления // Проблемы теории и практики управления. 2007. № 9. С. 63–68.

18. Царев Р. Ю. Преобразование атрибутов при многоатрибутивном принятии решения / Решетневские чтения : тез. докл. V Всерос. научн.-практ. конф. студентов, аспирантов молодых специалистов (12–15 ноября 2001, г. Красноярск). Красноярск, 2001. С. 119–120.

19. Ching-Lai Hwang, Kwangsun Yoon. Multiple Attribute Decision Making. Methods and Application. Berlin : Springer-Verlag, 1981. 255 p.

20. Stupina A., Volf P. Random point processes // САКС : материалы междунар. науч.-практ. конф. Красноярск, 2001. С. 273–276.

21. The multi-objective optimization of complex objects neural network models / V. S. Tynchenko, V. V. Tynchenko, V. V. Bukhtoyarov et al. / Indian Journal of Science and Technology. 2016. Vol. 9, No. 29. P. 99467.

22. Карасева М. В., Карцан И. Н., Зеленков П. В. Метапоисковая мультилингвистическач система // СибГАУ. 2007. № 3 (16). С. 69–70.

23. Карцан И. Н. Мультиверсионное программное обеспечение бортового комплекса управления с генетическим алгоритмом // Решетневские чтения : материалы XXI междунар. науч.-практ. конф. (08–11 ноября 2017, г. Красноярск). Красноярск, 2017. Т. 1. С. 372–373.

References

1. Volkov V. A. [Multiattribute selection of fault-tolerant software components]. *Vestnik universitetskogo kompleksa*. 2006, No. 8 (22), P. 208–211. (In Russ.)

2. Kovalev I. V., Savin S. V. [Optimal formation of redundant structure for fault-tolerant information systems]. *Issledovano v Rossii*. 2004, No. 7, P. 1123–1129. (In Russ.)

3. Kartsan I. N., Zhukov A. O., Platonov O. A., Efremova S. V. The hardware and software implementation of the adaptive platform for an onboard spacecraft control system. *Journal of Physics: Conference Series*. 2019, Vol. 1399, No. 3, P. 033071.

4. Kartsan I. N., Efremova S. V., Khrapunova V. V., Tolstopiatov M. I. Choice of optimal multiversion software for a small satellite ground-based control and command complex. *IOP Conference Series: Materials Science and Engineering*. 2018, Vol. 450, No. 2, P. 022015.

5. Kudymov V. I., Brezitskaya V. V., Zelenkov P. V., Kartsan I. N., Malanina Yu. N. Formation of optimal composition of the modules of single-function multiversion software for automated control system of the satellite communication system. *IOP Conference Series: Materials Science and Engineering*. 2018, Vol. 450, No. 5, P. 052009.

6. Kovalev I., Davydenko O. Interactive system for spacecraft technological control cycle construction. *Program and Abstracts of Int. Symposium SOR'96*. TU-Braunschweig (4–6 Sept. 1996). 1996, P. 195.

7. Stupina A. Realization of conventional pattern of random search methods in the space of Boolean variables. *Optimization Days*. 1997, P. 98–112.

8. Yunusov R. V. [Modeling of software architectures of automated control systems]. *Materialy Vserossiiskoi elektronnoi nauch.-tekhn. konf. "Upravlyayushchie i vychislitel'nye sistemy. Novye tekhnologii"* [Materials of the All-Russian Electronic Scientific and Technical Conf. "Management and computing systems. New Technologies"]. 2001, P. 60–61. (In Russ.)

9. Antamoshkin A., Stupina A. The random search algorithms in the space of Boolean variables. *Symp. OR'97*. Jena, 1997, P. 112–118.

10. Semenenko M. G., Kniazeva I. V., Beckel L. S., Rutskiy V. N., Tsarev R. Y., Yamskikh T. N., Kartsan I. N. How to use neural network and web technologies in modeling complex technical systems. *IOP Conference Series: Materials Science and Engineering*. 2019, Vol. 537, No. 3, P. 032095.

11. Semenko T. I. [Multiattributive approach to the formation of fault-tolerant software control systems]. *Successes of modern natural science.* 2004, No. 6, P. 32–33. (In Russ.)

12. Kovalev I. V., Tsarev R. Yu. [Combined method of multi-version spacecraft control software formation]. *Aviation and Space Instrument Engineering*. 2006, No. 9, P. 8–14. (In Russ.)

13. Tsarev R. Yu. *Mnogoatributivnoe prinyatie reshenii v mul'tiversionnom proektirovanii: monografiya* [Multi-attributive decision making in multiversion design : monograph]. Krasnoyarsk, IPC KSTU, 2005, 156 p.

14. Efremova S. V., Kartsan I. N., Zhukov A. O. An ordered ranking multi-attributive model for decision-making systems with attributes of control systems software. *IOP Conference Series: Materials Science and Engineering.* 2021, Vol. 1047, P. 012068.

15. Kartsan I. N. Models for Estimating the Reliability of the Software of an Onboard Control System. *Research journal of pharmaceutical biological and chemical sciences*. 2018, Vol. 9, No. 5, P. 2357–2367.

16. Tsarev R. Yu. [Analysis of qualitative and quantitative attributes in the design phase of fault-tolerant software systems]. *Control systems and information technologies*. 2006, No. 3(25), P. 95–101. (In Russ.)

17. Tsarev R. Yu. [Compensation model of multi-attribute decision making in the formation of management information systems]. *Problems of the theory and practice of management*. 2007, No. 9, P. 63–68. (In Russ.)

18. Tsarev R. Yu. [Attribute transformation in multi-attribute decision making]. *Materialy V Mezhdunar. nauch. konf. "Reshetnevskie chteniya"* [Materials V Intern. Scientific. Conf "Reshetnev reading"]. Krasnoyarsk, 2001, P. 119–120. (In Russ.)

19. Ching-Lai Hwang, Kwangsun Yoon. *Multiple Attribute Decision Making. Methods and Application*, Springer-Verlag, Berlin, 1981, 255 p.

20. Stupina A., Volf P. Random point processes. *Siberian Aviation and Space Salon*. 2001, P. 273–276.

21. Tynchenko V. S., Tynchenko V. V., Bukhtoyarov V. V., Tynchenko S. V., Petrovskyi E. A. The multi-objective optimization of complex objects neural network models. *Indian Journal of Science and Technology*. 2016, Vol. 9, No. 29, P. 99467.

22. Karaseva M. V., Kartsan I. N., Zelenkov P. V. [Meta-search multi-linguistic system]. *Vestnik SibGAU*. 2007, No. 3 (16), P. 69–70. (In Russ.)

23. Kartsan I. N. [The multiversion software of the onboard control complex with genetic algorithm]. *Materialy XXI Mezhdunar. nauch. konf. "Reshetnevskie chteniya"* [Materials XXI Intern. Scientific. Conf "Reshetnev reading"]. Krasnoyarsk, 2017, P. 372–373 (In Russ.).

Карцан Игорь Николаевич – доктор технических наук, старший научный сотрудник, Морской гидрофизический институт РАН; профессор, Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева; профессор, Севастопольский государственный университет. E-mail: kartsan2003@mail.ru.

Ефремова Светлана Владимировна – аспирант, Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева. E-mail: efremova_svet@sibsau.ru.

Kartsan Igor' Nikolaevich – Dr. Sc., Senior Researcher, Marine Hydrophysical Institute, Russian Academy of Sciences, Professor, Reshetnev Siberian State University of Science and Technology; Professor, Sevastopol State University. E-mail: kartsan2003@mail.ru.

Efremova Svetlana Vladimirovna – Postgraduate student, Reshetnev Siberian State University of Science and Technology. E-mail: efremova_svet@sibsau.ru.