CHAPTER III. DATABASE AND KNOWLEDGE SYSTEMS

Database replication method based on the task of (0,1)-knapsack

Dmytro Holubnychyi
Doctor of Philosophy in Technical Sciences,
Leading Research Officer, Air Force Science Center
Ivan Kozhedub Kharkiv National Air Force University
ORCID https://orcid.org/0000–0003–1719–7586

Viacheslav Tretiak
Doctor of Philosophy in Technical Sciences,
Leading Research Officer, Air Force Science Center
Ivan Kozhedub Kharkiv National Air Force University
ORCID https://orcid.org/0000–0003–2599–8834

Abstract. *Is proposed a parallel computation method based on the ranking approach to its solution the principle of optimization in the direction and strategies for cutting out unpromising decisions for fragmentation of network database data.*

Keywords: *database, fragmentation method, organization, replication, integer linear programming*

Introduction

Analysis [1–3] shows that each of the performance enhancements requires an analysis of the Relational Database (RDB) structure, the flow of solved tasks (Relational Database Management System (RDBMS) queries), and in some cases a detailed analysis of the use of individual RDB fragments (individual tables, their fields, table groups).

For example, managing an index structure requires determining the intensity of using one table and fields. restructuring involves changing the structure of one or more tables. Materialized Impressions (MPs) means a detailed analysis of the sequence of requests and the introduction of special tables for MPs.

Scientific groups under the leadership of leading scientists Y. Foster, Y. Gordon, R. Buya, E. Dilman, D. Tayne, T. Haye, R. Prodan, and also the research of the organization of efficient processing and fragmentation of data as a technology of the CCI of the SBG are actively engaged. A series of studies by Russian scientists

Gorobets V.V. was carried out. Zinkina SA, Kulba VV, Mikrin EA, Kosyachenko S. A., Plutonko A. D., Sokolinsky L. B., Tsvyaschenko E. V.

Setting the task of distributing fragments of the data of the network database

Let's consider the model of placement of fragments of the network database in the computing complex [5]. As the criterion of optimality, let's take the average amount of data sent through the data lines when the request is executed.

Considered n – the number of nodes of the network with an arbitrary structure; m–number of independent fragments of the stem database of data; c – this node of the network; K_j – j fragment of the network database, $j=\overline{1,n}$; F_i -an $i$object of this fragment $i=\overline{1,m}$; L_i – the memory of the node intended for the placement of fragments; b_j – number of request classes (for example, reading, adding, updating, deleting database records); λ_{ij}^k – intensity of requests k-that class to a fragment initiated in a node; α_{ij}^k – the volume of the request of k that class $\left(k=\overline{1,s}\right)$ to the fragment initiated in the node K_j; β_{ij}^k – the amount of data requested when the query of k that class $\left(k=\overline{1,s}\right)$ is performed to the fragment F_i that arrives at the node K_j; p is the number of tables in the network database; M_z z-th table of RBD, $z=\overline{1,p}$; m_z – number of rows of the table M_z; n_z – the number of columns in the table M_z; d_{ij}^z – the cell of the table M_z at the position (line i; column j), $i=\overline{1,mz}; j=\overline{1,nz}$; d_{ijz} – the amount of memory occupied by the cell d_{ijz}.

The volume of data sent in response to the request of that class to the fragment F_i initiated in the node is determined as follows $\left(\alpha_{ij}^k+\beta_{ij}^k\right)\left(1-x_{ij}\right)$. In this case, the values $x_{ij}\left(i=\overline{1,m}; j=\overline{1,n}\right)$ are determined as follows:

$$y_{ijz}^{\sigma}=\begin{cases}1, \text{ if the cell } d_{ijz} \text{ belongs to the fragment } F_{\sigma};\\ 0, \textit{ otherwise}.\end{cases} \quad (1)$$

Since the intensity generates the amount of data that needs to be transmitted, the total amount of data that is required to be transmitted over the communication channels between the nodes due to the functioning of the distributed system over a unit of time is determined by the formula:

$$S=\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{s}\lambda_{ij}^k\left(\alpha_{ij}^k+\beta_{ij}^k\right)\left(1-x_{ij}\right). \quad (2)$$

If we put that $\lambda=\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{s}\lambda_{ij}^k$ the target function of the problem of optimal distribution of fragments over the nodes of the computing complex takes the form:

$$V=\frac{1}{\lambda}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{s}\lambda_{ij}^k\left(\alpha_{ij}^k+\beta_{ij}^k\right)\left(1-x_{ij}\right). \quad (3)$$

The lower the value, the higher the speed of service queries. All messages arriving at the node's input queues are divided into two types: type 1 – messages that make up queries for processing which the necessary fragments are not contained in the database of the computing complex node, and the responses to these que-

ries; type 2 – messages that make up requests for which the necessary fragments are stored in the database of the corresponding node of the computing complex. In this case, the request of type 1, arrived for its service to the remote node of the computing complex, turns into a request of type 2.

Since each fragment $F_i\left(i=\overline{1,m}\right)$ must be located in at least one node of the computing complex, then the condition

$$\sum_{j=1}^{n} x_{ij} \geq 1, i=\overline{1,m}. \tag{4}$$

In addition, the volume of the local database of each node should not exceed the amount of memory of this node intended for the placement of fragments:

$$\sum_{i=1}^{m} L_i x_{ij} \leq b_j, j=\overline{1,n}. \tag{5}$$

The total volume of all fragments should be equal to the total volume of all tables in the network database, i.e.

$$\sum_{i=1}^{m} L_i = \sum_{z=1}^{p}\sum_{i=1}^{m_z}\sum_{j=1}^{n_z} w_{ij}^{z} \tag{6}$$

At the same time, the volume of the fragment is equal:

$$L_\sigma = \sum_{z=1}^{p}\sum_{i=1}^{m_z}\sum_{j=1}^{n_z} w_{ij}^{z} y_{ijz}^{\sigma} \tag{7}$$

Thus, the problem of allocating fragments of the data of the network database to the nodes of the computing complex is to determine the values x_{ij} of variables where, which $x_{ij}=\{0;1\}\left(i=\overline{1,m}; j=\overline{1,n}\right)$ satisfy conditions (4) (5) and give the minimum of the linear function (3). The obtained mathematical model is intended for solving the problem of integer linear programming.

Method of cutting off unpromising options for problem (0.1) knapsack

The method of cutting off unpromising options for problem (0.1) knapsack put a set of strategies $\{L_w\}$, whose application to the generalized procedures $A0$ lead to the construction of algorithms for solving this problem of integer linear programming with Boolean variables.

The simplest strategy of clipping $L1$ in the formation of $r=r+1$ paths of the next rank in sets m_{sp}^{r+1} based on the procedure $A0$, is the allocation m_{sj}^{r} of paths of maximum length by weight of functional sj whose lengths by weight restrictions do not exceed the magnitude of b_i, $d_a\left(\mu_{sj}^{r}\right)_i \leq b_i$. In this case, the paths that satisfy the properties of v are called μ_{sj}^{r} paths whose lengths by weight do not exceed the value of b_i. Then, recurrence relations, corresponding to the strategy $L1$, takes the form:

$$\mu_{sp}^{r+1} = \max_{c_j}\left\{\mu_{sj}^{r}\bigcup(j,p)\right\},\ p=\left(\overline{r+1,n}\right),\ j=\left(\overline{r,n}\right), \tag{8}$$

Using the $L1$ strategy does not always provide the optimal solution to the problem.

The occurrence of the screening of paths by the ratio corresponding (8) to the optimal solution of the problem is due to the fact that the paths with a higher value for the functional can gain more length by weight restrictions. Therefore, at some rank there is a situation where, due to restrictions, the paths of the next ranks in the graph $D\Delta$ cannot be constructed, and paths with smaller values of length by weight of the functional having smaller length by weight of constraints discarded by strategy $L1$.

On basis of this, paths of higher rank could be obtained, and, at the expense of this, a greater value by weight of the functional.

Denote $\overset{**r=q}{\mu}_{sj=\gamma}$ way rank $r=q$ with a maximum length $d_c\left(\overset{**r}{\mu}_{sj}\right)$ of ways $\left\{\overset{*r<q}{\mu}_{sj}\right\}$ rank $r<q$, corresponding to local extremes caused by $A0$ procedure using a strategy of $L1$. Then formulate a rule for elimination $L2$-dimensional problem based on the following theorem.

Theorem 1 column $D\Delta$ there is no way that satisfies properties v, rank $r<q$ for which the inequality $d_c\left(\overset{*r<q}{\mu}_{sj}\right)<d_c\left(\overset{**r=q}{\mu}_{sj=\gamma}\right)$.

Theorem 1 follows a strategy clipping $L2$, allowing mentioned functional way $\overset{**r=q}{\mu}_{sj}$ used as an upper estimate for elimination pathways in sets $m_{sj}^{r<q}$ column $D\Delta$. Strategy $L2$ used in constructing multistage algorithms.

Without prejudice strategies $L1$ and $L2$, you can enter additional sets a clipping paths m_{sj}^{r}. Representing strategy $L3$, which is based on the properties of the graph $D\Delta$ and is as follows. From the structure of the graph $D\Delta$ shows that for each of its vertices j, which corresponds to the region Ω_j weight γ_j is the sum of the coefficients sj functional (5), due to the rule:

$$\gamma_j=c_{j+1}+c_{j+2}+...+c_n,\ \gamma_n=0;\ j=\overline{(1,n-1)}, \tag{9}$$

represents the upper bound value increasing importance in local extremes Ω_j on all these levels.

Reducing the number of shaped paths μ_{sj}^{r} plural m_{sj}^{r} can be achieved if the conditions caused by the following fairly obvious statement 2.

Statement 2. If the total path length $d_c\left(\mu_{sj}^{r}\right)$ a subset of r m_{sj}^{r} w and in weight γ_w. This summit has received less than the maximum length $d_c\left(\mu_{sj=p}^{r}\right)$. Then the next rank ways that are based on μ_{sj}^{r}. Can not determine the optimal solution.

The validity of the assertion 2 follows from the fact that the strategy $A0$ procedure $L1$ selects global extreme of all the local and the maximum length by way of extension μ_{sj}^{r} there is less of a local extreme. Thus, test conditions:

$$d_c\left(\mu_{sp}^{r}\right)+\gamma_p<\max_{\{c_j\}}\left\{d_c\left(\overset{*r}{\mu}_{sp}\right)\right\}, \tag{10}$$

where $d_c\left(\mu_{sp}^{r}\right)$ – path length μ_{sp}^{r} the top rank pr weight c_j; avoids this path of further analysis as futile, if the condition is fulfilled. This strategy is based verification $L3$.

As a result of the procedure $A0$ strategies $L1$, $L3$ are set m_{sj}^r. For which $r > q$, and for effective filtration unpromising paths must be functional (5) assessment valuesin fields Ω_j, $j = \overline{(1, n^2/2)}$ at $r > q$. For this we introduce the strategy of choice from a plurality of paths $L4$, based procedure, which gives a maximum possible paths rank r in column $D\Delta$. It can be implemented if $D\Delta$ determine the shortest path weight restrictions based procedures $A0$ s between the top and all the other vertices of the graph $D\Delta$, and the recurrence relation (6) for a one-dimensional problem takes the following form:

$$\mu_{sp}^{r=r+1} = \min_{a_{ij}} \left\{ \mu_{sp}^r \bigcup (j.p) \right\}, p = \overline{(r+1, n)}, \; j = \overline{(r, n)}, \; i = 1. \tag{11}$$

It is easy to see that with this strategy, the formation of plural paths m_{sj}^r proposition 2 remains valid and elimination pathways in sets m_{sj}^r you can use the condition (11).

Implementing procedures $A0$ based strategies $L3$, $L4$ allows us to formulate more unpromising strategy $L5$ clipping paths in sets m_{sj}^r, due to the following theorem 2.

Theorem 2. If the set $m_{sj}^r, j = \overline{(r_{\max}, n)}$, the highest rank $r_{\max}$ built $A0$ procedure strategies $L3$, $L4$, choose the path μ_{sw}^r with the greatest value by weight functional (5) in case of repeated deci- sions outbound enough ways to form the first rank $r = 1$ in the set $m_{sp}^r, p = \overline{(1, k)}$, where:

$$k = w - r_{\max} + 1, \tag{12}$$

and w – number set m_{sp}^r with a maximum length by weight functional due to the first solution to the problem.

In a theorem 3 we formulate a rule that allows multiple m_{sj}^r provide a way $\overline{\mu}_{sj}^r$ length of $d_c\left(\overline{\mu}_{sj}^r\right)$, in relation to which all paths of length less than the value of the functional weight can be excluded from the analysis as unpromising.

Theorem 3. If you sort the set $m_{sj}^r, j = \overline{(r_{\max}, n)}$ rank r vectors in descending order by weight functional c_j, then the way in which the length $d_c\left(\mu_{sj}^r\right)$ less than the length $d_c\left(\min_{a_{ij}}\left(\mu_{sj}^r\right)\right)$ can not determine the optimal solution.

This theorem defines the corridor: in one-dimensional corridor set m_{sj}^r the set $m_{sp}^{r=r+1}$ we mean a set of paths μ_{sj}^r, who are between the upper limit set m_{sj}^r and its lower limit satisfying properties ν plural $m_{sp}^{r=r+1}$. The upper limit is determined by the maximum $d_c\left(\mu_{sj}^r\right)$ and the bottom – Theorem 3.

From concept corridor can propose the following strategy $L7$, which represents another strategy of choice routes r is set to from m_{sj}^r rank r in the set $m_{sp}^{r=r+1}$, $p = \overline{(1, n)}$ next rank choose paths that satisfy properties ν is the maximum weight cj functional and minimal weight restrictions aij, corresponding recurrence relations (9), (12).

A number of $\{L_w\}$ clipping strategies have been considered for the one-dimensional of integer linear programming with boolean variables. We formulate clipping strategies for m-dimensional problems.

If the solution m-dimensional problems of the most simple strategy is the strategy of choice $L10$, based on the recurrence ratio (9). The principle of forming paths in the identical strategy sets $L1$, except when necessary to carry out checks m times.

The combination strategy with the strategy $L7$, L3 in the case of m-dimensional problem $L11$ form a selection rule, which is to set out m_{sj}^{r} the set $m_{sp}^{r=r+1}$, $p=\overline{(1,n)}$. Next rank choose ways which satisfy properties ν is the maximum weight and minimum functional c_j for each i-th weight restrictions $a_{ij}, i=\overline{(1,m)}$.

The first corresponds recurrence relations (9) and the second is described by the following equation (13), based on the ratio (12):

$$\mu_{sp}^{r=r+1}=\min_{a_{ij}}\left\{\mu_{sj}^{r}\bigcup(j.p)\right\}, p=\overline{(r+1,n)},\ j=\overline{(r,n)}. \qquad (13)$$

Enter by analogy with the concept of one-dimensional corridor concept m-dimensional corridor set m_{sj}^{r}.

Let subset of vectors that must remain in the set m_{sp}^{r} after filtering through m_{sp}^{rk}. Recall that the formation of a plurality of paths m_{sp}^{r} always carried out so that the length of the first functional weight are more functional length of the weight of the second and so on. e., that $d_c(\mu_k)\geq d_c(\mu_{k+1})\geq ...\geq d_c(\mu_n)$.

Procedure $A0$ strategies $\{L_w\}$ Kw and regulations can serve as a basis for constructing approximate and exact algorithms for solving the problem of the knapsack is to get different versions $A0$ procedures depending on the combinations used rules clipping $\{L_w\}$ unpromising paths insets μ_{sj}^{r} by applying the principle of optimization direction.

Approximate algorithms for solving the problem integer linear programming with boolean variables

The main qualifying features are the number of steps and the accuracy of the steps of solving the problem. By the first feature, the algorithms are divided into one-stage and multi-stage. This division means that the initial task can be solved one (one-stage) or several times (multi-stage) by different algorithms, the list of which is given in parentheses. The second sign determines the error of the received admissible solution of the problem by the weight of the functional from the optimal solution. If the algorithm is fundamentally unable to give an exact decision about the weights of the functional, then we will call such an algorithm approximate.

Algorithm $A1$. Consideration of approximate algorithms start with a one-dimensional problem. The simplest algorithm $A1$ is that the procedure is based on the $A0$ and implements strategies $L1$ and $L3$. His step description is as follows.

STEP 1. From the top of the set s built paths $m_{sj}^{r=1}$, $j=\overline{(1,n)}$ first rank r, satisfying the properties and determined to set $m_{sj}^{r=1}$ path of maximum length $\left\{\mu_{sj}^{*r}\right\}$ weight functional s_j. For each node j determined weightγj rule (10).

STEP 2. excluded paths $\{\mu_{sj}^r\}, p=\overline{(r,n)}$ plural m_{sj}^r current rank r, the length of which $d_c\left(\mu_{sp}^r\right)$ satisfying inequality (11).

STEP 3. Formed a set of paths $m_{sp}^{r=r+1}$, $p=\overline{(1,n)}$. next rank satisfying properties based on a plurality of paths m_{sj}^r previous rank based on recurrent ratio (9). In educated sets $m_{sp}^{r=r+1}$ distinguish longest path $\left\{\begin{matrix}* r=r+1 \\ \mu_{sp}\end{matrix}\right\}$ If several ways will be the maximum length, the path chosen among them with a lower value length weight restrictions a_{1j}.

STEP 4. Checks whether all paths set next $r=(r+1)$ rank is empty. If so, proceed to step 5; if not, check $r=(n-1)$. In the case of equality, go to step 5, otherwise increase r 1 and perform step 2.

STEP 5. Select the plural maximum path length and algorithm $A1$ terminates.

Getting close decision is due to the strategy of choice $L1$.

Algorithm $A2$. Another strategy is the strategy of choice routes $L4$, using an algorithm that allows $A2$ obtained by the maximum possible rank r in column $D\Delta$. It can be implemented if the D to determine the shortest path weight restrictions based procedures $A0$ between the top and the rest of s vertices D, using the recurrence relations (12) and, of course, clipping rule $L3$. Step by step description of the algorithm $A2$ is as follows.

STEP 1. From the top of the set s built paths $m_{sj}^{r=1}, j=\overline{(1,n)}$ first rank r, satisfying the properties and determined to set $m_{sj}^{r=1}$ path of maximum length $\left\{\begin{matrix}* r \\ \mu_{sj}\end{matrix}\right\}$ weight functional s_j. For each node j determined weightγj rule (10).

STEP 2. Excluded paths a plurality m_{sj}^r Current rank r, the length of which $d_c\left(\mu_{sp}^r\right)$ satisfying inequality (11).

STEP 3. Formed a set of paths $m_{sp}^{r=r+1}$, $p=\overline{(1,n)}$. next rank satisfying properties based on a plurality of paths m_{sj}^r previous rank, who have a minimum value m_{sj}^r length weight restrictions based on recurrence relation (12). In educated sets $m_{sp}^{r=r+1}$ distinguish longest path $\left\{\begin{matrix}* r=r+1 \\ \mu_{sp}\end{matrix}\right\}$

STEP 4. Checks whether all paths set next $r=(r+1)$ rank is empty. If so, proceed to step 5; if not, check $r=(n-1)$. In the case of equality, go to step 5, otherwise increase r 1 and perform step 2.

STEP 5. Select the plural path length and the maximum number tops last w, belonging to this path, then the algorithm $A2$ terminates.

Algorithm $A3$. Based on the concept of corridor and $L7$ accordance with the strategy, the following approximate algorithm $A3$ solution which is to set out m_{sj}^r a plurality of rank r $m_{sj}^{r=r+1}, j=\overline{(r+1,n)}$ be chosen next rank up to two ways that satisfy properties ν.

The first is the path of maximum weight in functionality (5) (relative recurrence relation (9), and the second – the path with the least weight restriction on $\{a_{1j}\}$ (relatively recurrence relation (12) that appropriately change the step 3 algorithm $A1$. In another step description of the algorithm $A3$ coincides with the description of the algorithm $A1$. Loss optimum solution due to its falling into the corridor.

Algorithm $A4$. For multidimensional problem can also be similarly effective to construct approximate algorithms. This algorithm $A4$ is based on the strategy of $L10$, who set the current builds in rank rank next set $r=(r+1)$ path with a maximum length weight functional $\{c_j\}$.

This temporal complexity of the algorithm increases 2^m as necessary to execute each vector operations of addition m and m comparisons to ascertain whether satisfying vector corresponding to this path restriction or not. So step and $A4$ for algorithm description coincides with the algorithm $A1$.

Algorithm $A5$. Application strategy $L11$ allowed to build algorithm $A5$, the essence of which is the choice of each set m_{sj}^r Current rank r one way, maximum weight and functional, at worst, m routes, minimum weight limit satisfying properties v. Then, unlike algorithm $A1$, $A5$ algorithm step article describes changes to the following:

STEP 1. From the top of the set s built paths $m_{sj}^{r=1}, j=\overline{(1,n)}$ first rank r, satisfying the properties and determined to set $m_{sj}^{r=1}$ path of maximum length $\left\{\mu_{sj}^{*r}\right\}$ weight functional. For each node j c_j weight is determined by the rule (10).

STEP 2. excluded paths a plurality m_{sj}^r Current rank r, length which $d_c\left(\mu_{sp}^r\right)$ satisfying inequality (11).

STEP 3. Formed a set of paths $m_{sp}^{r=r+1}$, $p=\overline{(1,n)}$. next rank satisfying properties based on a plurality of paths m_{sj}^r previous rank based on recursive relationships (9), (13). In educated sets $m_{sp}^{r=r+1}$ distinguish longest path $\left\{\mu_{sp}^{*\,r=r+1}\right\}$ If you find some ways to the maximum length, the path chosen among them with a lower value length weight limit $\{a_{ij}\}$. If you find a number of ways with a minimum value for the i-th restriction, the chosen one has more weight functional length.

STEP 4. Checks whether all paths set next $r=(r+1)$ rank is empty. If so, proceed to step 5, if not, check $r=(n-1)$. In the case of equality, go to step 5, otherwise increase r on 1 and perform step 2.

STEP 5. Select the plural maximum path length and $A5$ algorithm terminates.

This algorithm has one uncertainty in the case when multiple vectors with maximum length by weight of the functional and with different values by weight of constraints are found in the set, ie, for example, by the weight of the first restriction more than the second vector, by the weight of the second restriction, by the weight of the third – again the second, etc.

Therefore, in order to avoid this uncertainty in the equality of lengths by weight of the functional path is chosen with a smaller value by weight of all restrictions. If this is not possible, then any is selected. It may be that the rejected path will belong to the optimal path. This fact is one of the reasons for the $A5$ algorithm approximation.

To check the correctness and accuracy of the solution by the developed algorithms $A1$ – $A5$ a modified Balash algorithm [3] is chosen as a standard, which takes into account the requirements of the coefficients in the functional. In the future, by D we will understand exactly this algorithm. In the course of

solving the test problems, coefficients were generated using the random number sensor in functional in the range [1÷100] and in the constraints of condition in the range [1÷10]. The choice of other ranges for the functional changed only its absolute value, but did not affect the parameters of the algorithms on average. Changing the range in constraints only affects the rank of the path (path rank is the number of vertices of the graph $D\Delta$ that make up that path).

The structure of the study of each algorithm was as follows. 1000 test problems were solved with given input parameters m and n by the selected algorithm. The performance of the algorithms was calculated during the decision. The values obtained were shown in the least squares approximated graphs.

All algorithms are divided into two groups. The first group consists of one-dimensional algorithms $A1$, $A2$, $A3$, which solve problems ILP with BV. The second group consists of m-dimensional algorithms $A4$, $A5$, that solve the problem.

Research algorithms of the first group. As it turned out, the quantitative values of the selected indicators depend significantly on the rank $\bar{r}$ f the resulting solution, which determines the number of units in the optimal solution. Figures 1 and 2 confirm this, depicting the time dependence of N_{on} algorithms on $\bar{r}$ with fixed values of input parameters n and m = 1.

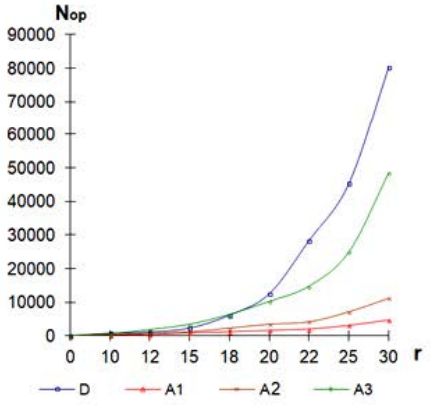

Figure 1. – The dependence of N_{op} on $\bar{r}$ when $n = 15$

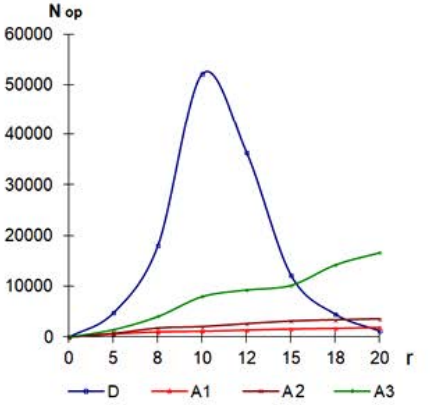

Figure 2. – The dependence of N_{op} on $\bar{r}$ when $n = 20$

The (Fig. 1 and 2) show that the range of variation $\bar{r}$ can be divided into three zones: zone 1 – $\bar{r} = [0 : n/3]$; 2 zone – $\bar{r} = [n/3 : 2n/3]$; zone 3 – $\bar{r} = [2n/3 : n]$. In the first zone, algorithm $D\Delta$ finds the solution quickly, because at the expense of probing the branches of the decision tree corresponding to a single branch are very effectively cut off. Similarly, the fast solution in 3 zone is explained, the vectors in which consist of a large number of unit. The manifestation of all the exponential complexity of algorithm $D\Delta$ occurs in the second zone. For algorithms $A1$, $A2$, $A3$, the increase in the number of N_{on} with

increasing $\bar{r}$ is determined by the number of local areas W that need to be processed. According to the expression the number of regions in the W column $D\Delta$ does not exceed $n^2/2$, so $r \to n$ nop increase in the number does not exceed $c \cdot n^2$, where c = const.

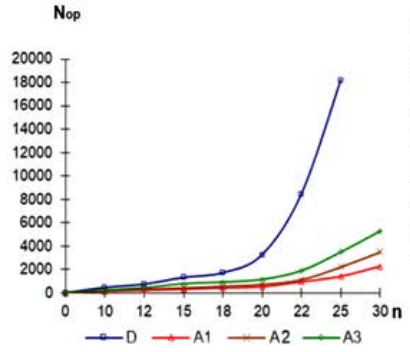

Figure 3. – The dependence of N_{op} on n for zone 1

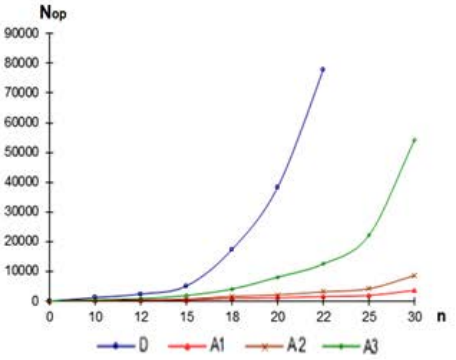

Figure 4. – The dependence of N_{op} on n for zone 2

Thus, for objective comparison of algorithms it is necessary to specify in which zone they belong, that is, what percentage of units (or zeros) contains the optimal solution. According to the division into zones, we plot the dependence of the number N_{on} on n for each zone. These dependencies for zones 1, 2 and 3 are shown in (Fig. 3–5). Figures 3, 4, 5 show that in order to compare the time indices of the algorithms it is necessary to clearly indicate the belonging to the conditional zones, since the absolute value of these indicators in the zones differs significantly. The hit of a solution of a problem in any zone is precisely determined by the ratio of the range of change of coefficients a_{1j} and the given range for b_1 on the right-hand side of inequality.

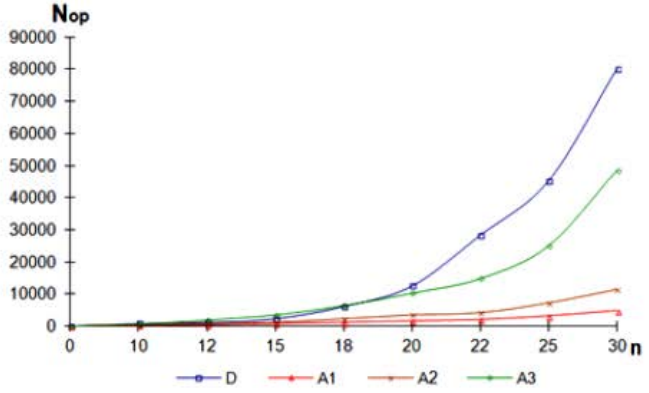

Figure 5. – The dependence of N_{op} on n for zone 3

The amount of RAM V required depends on the specific software implementation. The quantitative values of this indicator, of course, can be changed and directly depend on the skill of the programmer. However, the appearance of the curves does not change due to the nature of the processes they describe.

We show that the amount of RAM required to store the set of m_{sj}^{r} vectors of the graph $D\Delta$ also depends on the average rank of the optimal solution of the test problem. The dependence $V = f(n)$ for the algorithms D, $A1$, $A2$, $A3$ is plotted (figures 6 and 7).

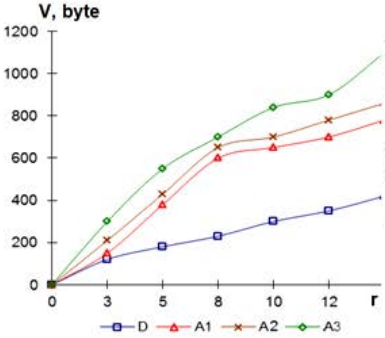

Figure 6. – Effect $\bar{r}$ the V for $n = 15$

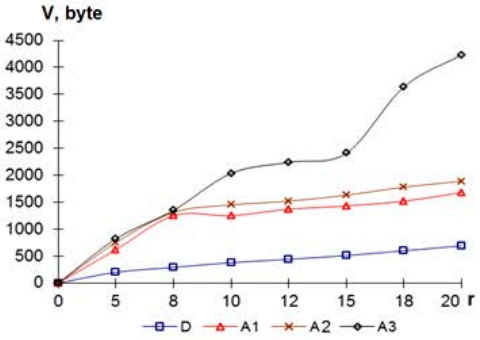

Figure 7. – Effect $\bar{r}$ the V for $n = 20$

The smaller value of the required amount of RAM for algorithm D is explained by the fact that in case of successful probing by the relations (7) and (8) of subsets of alternatives, a new task, and therefore a new record in memory, is not formed because the free list is adjusted vertices for the selected task.

Only the formation of a new branch, which corresponds to step 6 of the algorithm, requires the amount of RAM. For algorithms $A1$, $A2$, $A3$ the maximum number of processed vectors stored in the second area.

Therefore, more storage is required than the first zone. The maximum value of the cost is reached in the third zone, since in this case the difference in the weight of the functional within the sets is insignificant, which leads to inefficient screening of vectors according to the strategy $L3$.

The dependence of the value of V on the size of the input parameter n also does not exceed $c \cdot n^2$. The results of this study are shown in (Fig. 8–10), corresponding to zones 1–3.

The temporal complexity of the $A1$ and $A3$ algorithms is easy enough to estimate theoretically. The analysis of each $\vec{x}$ vector involves one addition operation and one comparison operation, that is, two elementary operations.

The maximum number of vectors in the worst case will be in the second rank and is $n^2/2$. The number of such ranks in the graph $D\Delta$ is n. Therefore, the time complexity of Algorithm $A1$ will be $O\left(2n \cdot \frac{n^2}{2}\right) = O(n^3)$, to store vectors in the Ω-regions at all ranks will require memory of no more than $O(n^3)$.

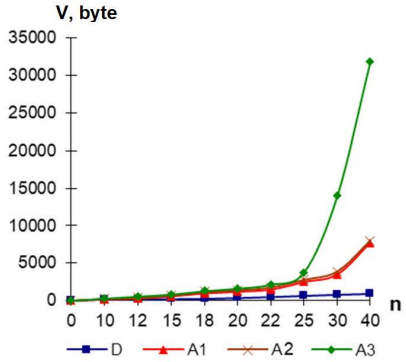

Figure 8. – The dependence of V on n for zone 1

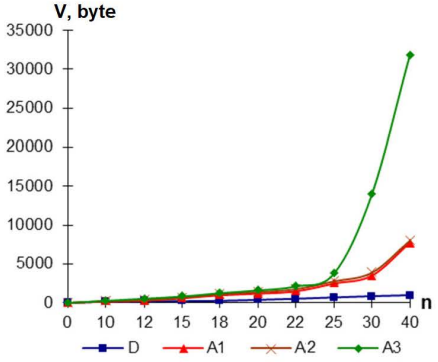

Figure 9. – The dependence of V on n for zone 1

The estimate of the time complexity of the $A2$ algorithm coincides with the estimate for the $A1$ algorithm.

Let estimate the time complexity of algorithm $A3$. The time complexity of algorithm $A1$ is $O(n^3)$. However, in algorithm $A3$ each set will contain, after isolation of the corridor and filtering in the worst case, two ways, which will increase the time complexity of algorithm $A3$ twice and will be $O\left(2 \cdot n^3\right) \approx O\left(n^3\right)$.

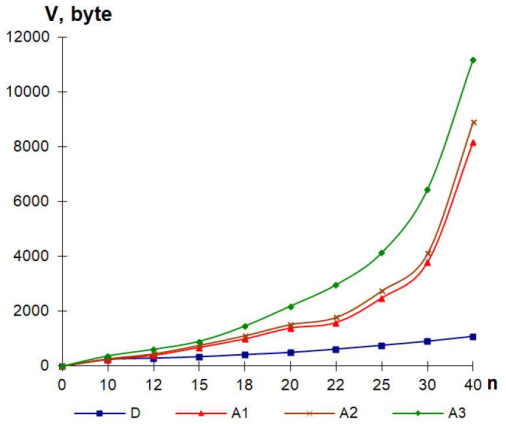

Figure 10. – The dependence of V on n for zone 3

Therefore, hardware costs for storing vectors will also double and add up to $O\left(2 \cdot n^3\right) \approx O\left(n^3\right)$.

One of the important characteristics of approximate algorithms is the relative error Δf, which it gives in the case of solving the problem. From these algorithms approximate are $A1$ and $A3$. The relative error depends on the area in which the optimal solution falls. However, obtaining inaccurate solutions by $A1$ and $A3$ algorithms for different zones proved to be a problematic task on average.

Thus, for zone 1, the exponential growth of vectors in the graph $D\Delta$ encounters a constraint, and therefore in optimal sets the optimal paths will dominate the functional weight In zone 3, the optimal path will be along the higher vertices of $D\Delta$, which will not allow it to be cut off.

Therefore, Figures 11, 12 show the variation of the error f and the K_n factor depending on the dimension n only for the second zone.

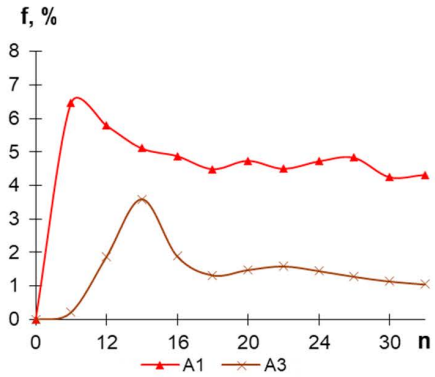

Figure 11. – The dependence of f on n for zone 2

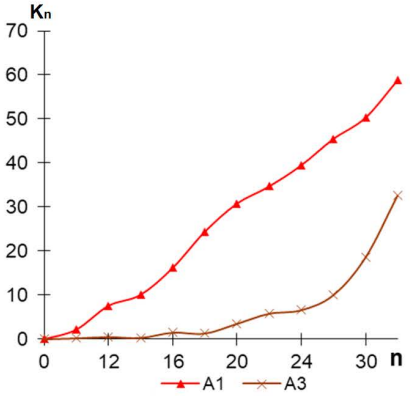

Figure 12. – The dependence of K_n on n for zone 2

As can be seen from (Figures 11, 12) for both $A1$ and $A3$ algorithms the average error f stabilizes: about 5% for the first algorithm and about 2% for the second at $n > 25$.

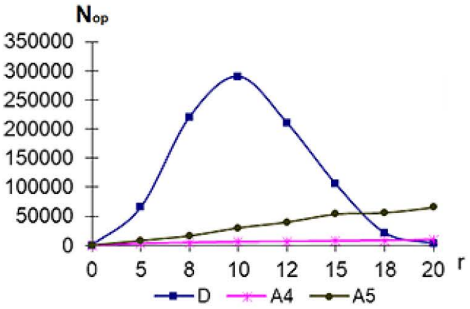

Figure 13. – The dependence of N op on $\bar{r}$ for $n = 20$ and $m = 5$

Research of algorithms of the second group. Let us evaluate the properties of the algorithms of the second group $A4$, $A5$ according to the selected performance indicators. The algorithms $A4$ and $A5$ are approximate. As for one-dimensional tasks, the values of these indicators depend on, $\bar{r}$ which is confirmed by the graph in (Figure 13). The curves in this figure are identical in shape to the corresponding indices of one-dimensional algorithms. Therefore, the division by zones is valid and

for multidimensional algorithms; the dependencies of the selected metrics on n for each zone also coincide with the corresponding dependencies for the one-dimensional algorithms. Due to the need to perform m times of verification of condition, the time complexity of the algorithm $A4$ will be $O\left(2 \cdot m \cdot n^3\right) \approx O\left(m \cdot n^3\right)$.

The time complexity of the algorithm $A5$ due to the chosen strategy $L11$ in m times larger than the time complexity of the algorithm $A4$ and is $O\left(m \cdot n^3 \cdot m\right) = O\left(m^2 \cdot n^3\right)$, since it is necessary to choose from each set according to relation (13), in the worst case, m more vectors. Therefore, the amount of memory hardware used to store the paths will also increase by a factor of m to $O\left(m \cdot n^3\right)$.

When m = 5, the errors of the $A4$ and $A5$ algorithms were investigated. The dependence of f and Kn on n and m for zone 2 are plotted in (Figure 14).

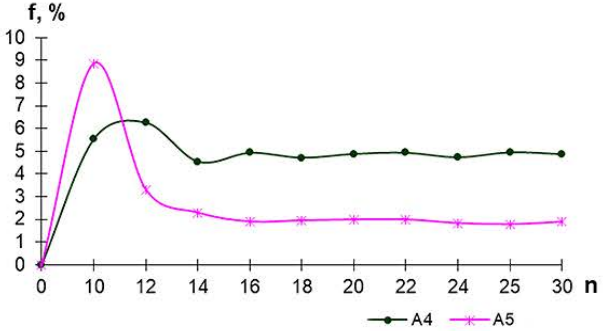

Figure 14. – The dependence of f on *n* for zone 2

Comparing Figure 11 and Figure 14, it can be seen that the relative error f does not depend on m, but depends only on the algorithm selected. Thus, it can be argued that the $A4$ algorithm has an error of 5% and the $A5$ algorithm of 2%.

Conclusions

An official model approach for solving integer programming with boolean variables has been developed Put $A0$ standard procedure that determines the rules of formation sets in box recovered $D\Delta$ and selecting options for solving the problem.

$A0$ procedures clipping for non-promising solutions are formulated based on a strategy that avoids the complexity of NP programming problems with integer boolean variables and provides an approximate solution

References:

1. Listrovoy S. V., Golubnichiy D. Yu., Listrovaya E. S. Solution method on the basis of rank approach for integer linear problems with boolean variables. Engineering Simulation. 1999. – Vol. 16. – P. 707–725.
2. Listrovoy S. V., Tretjak V. F., Listrovaya A. S. Parallel algorithms of calculation process optimization for the boolean programming problems. Engineering Simulation. 1999. – Vol. 16. – P. 569–579.

3. Tretiak V.F., Pashneva A.A. Optimizatsiya struktury khranilishcha dannykh v uzlakh seti infokommunikatsionnoi seti oblachnogo khranilishcha // Sistemy navigatsii i svyazi Poltavskogo natsional'nogo universiteta imeni Yuriya Kondratyuka.– № 4(44), 2017.– P. 122–128.