Parallel Approach of the Algorithm of Finding the Optimal Solution of the Transport Problem by the Method of Potentials

Lesia Mochurad[0000-0002-4957-1512], Nataliya Boyko[0000-0002-6962-9363],

Maryana Bortnikova[0000-0002-2346-8007]

Lviv Polytechnic National University, Lviv79013, Ukraine

lesia.i.mochurad@lpnu.ua, nataliya.i.boyko@lpnu.ua,
mariana.H.Bortnikova@lpnu.ua

Abstract. This paper analyzes the implementation of a parallel algorithm for finding the optimal solution of a transport problem by the potential method using OpenMP technology on dual- and quad-core processor systems. The results are obtained, which indicate the possibility of further optimization of the computational process of finding the optimal solution of the transport problem by varying the number of parallel streams and processor cores of the computer. A number of numerical experiments have been performed to confirm the effectiveness and reliability of the approach witch proposed in the paper.

Keywords: Multithreading, finite difference, linear programming, parallelization, multicore, optimal plan, parallel algorithm, Amdal law, multi-core, OpenMP software standard

1 Introduction

Today in economic science, the mathematical model comes to the fore as an effective tool for researching and predicting the development of economic processes and phenomena.

A special place among the economic-mathematical models is the transport problem [1-3].

The application of this task is making managerial decisions gives the opportunity to rationally distribute the funds and time of the enterprise, which has a positive impact on its development.

Under the name “transport problem”, a wide range of problems is combined with a single mathematical model [4].

Classic transport problem - the problem of the most economical plan of transportation of homogeneous product or interchangeable products from the points of production to the points of consumption, is most often found in practical applications of linear programming [2, 5].

It should be noted that many problems in their mathematical formulation are reduced to a transport problem.

Thus, with this method you can find the optimal organization of work in the planning of construction sites, the distribution of a given volume of work between different people, the delivery of goods and more.

Finding the optimal solution to a transport problem can find application in different areas:

- in economic
- in logistics planning;
- in the educational process;
- when developing databases;
- in programming.

Such a wide range of applications testifies to the relevance of the subject under study task.

The rapid expansion of the spheres of application of the methods of mathematical modeling and optimization, which has been happening in recent years, creates the need for solving new classes of problems of increased complexity and dimension [6, 7].

The article proposes the application of a parallel algorithm for finding a solution to a transport problem with current trends in the development of a computer system.

The main advantage of parallel algorithms over sequential ones is the speed of execution [8]. And given the transience of the modern world, there is a need to turn sequential algorithms into parallel ones because they speed up program execution time.

Therefore, the purpose of this article is to parallelize the algorithm of finding the optimal solution of a transport problem by the potential method by using OpenMP parallel programming technology, that is, to take maximum account of current trends in the development of multi-core processors.

2 Review of the Literature

At the present stage of development of society and economy, specialists must constantly improve the principles and methods of managing complex economic systems.

In these circumstances, the manager must have the basic principles and techniques of mathematical programming, and be able to apply them in practice.

Thus, a wide range of economic problems associated with managing complex processes require the use of modern methods of optimization and management decision-making.

It is the application of the transport problem to solve the distribution tasks in various fields of economic activity that enables to obtain the most effective results and to put them into practice successfully.

The transport problem (1) - (3) [9] is a linear programming problem that is used to determine the most economical plan for transportation of homogeneous products from suppliers to consumers.

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{1}$$

$$\sum_{i=1}^{m} x_{ij} = b_j, \ j = 1,2,..., n, \tag{2}$$

$$\sum_{i=1}^{m} x_{ij} = a_i, \ i = 1,2,..., m, \qquad x_{ij} \geq 0 \tag{3}$$

Here x_{ij} - the volume of production, c_{ij} - the rate of delivery of products from the i - supplier to the j - consumer, b_j - the needs of consumers for products, a_i - the inventory of products from suppliers.

There are two steps to solving a transport problem:

1) building a reference plan [1]:

- method of the northwest corner;
- the least cost method;
- the Vogel method.

2) search for the optimal plan [10]:

- the Hungarian method;
- potential method.

When solving a transport problem, you must have an initial reference plan.

The work of finding this plan was implemented using the least cost method.

The rules for finding an initial reference plan for a transport problem by the least cost method differ from the rules for finding such a plan diagonally by the sequence of cell selection that you want to fill.

According to the method of least cost, the cell with the least cost of transporting a unit of cargo from the supplier to the consumer is selected first.

If there are several such cells, then we choose the one for which the quantity of transportable cargo is the largest.

After constructing the initial reference plan, each of the methods in the table must be filled in $(m+n-1)$ by cells, because the rank of the matrix of the transport task constraint system is equal $r = m+n-1$, where m - the number of suppliers, n - the number of consumers.

If the number of cells filled is less $(m+n-1)$, then the plan is degenerate.

Then it is necessary to fill in an appropriate number of empty (nonbasic) cells, recording in them "zero transportation", and to consider these cells basic.

However, when the number of cells filled exceeds $(m+n-1)$, the initial reference plan is not properly constructed and is not the reference one.

Next we focus on a detailed analysis of the potential method to obtain the optimal solution.

1. Optimality criterion of the reference plan by the method of potentials.

We already know the methods of finding the initial support plans of a transport problem, but we do not know if these basic plans are optimal, ie, giving the lowest total cost of transportation of all cargo from suppliers to consumers.

The reference plan is tested for potential optimality.

According to each supplier A_i we put the potential u_i, a to each consumer $B_j - v_j$.

Optimality criterion for the transport plan reference plan: if for some reference plan (x_{ij}) of a transport problem there are such potential numbers u_i and v_j that for basic cells the equality $u_i + v_j = c_{ij}$, is fulfilled and for non-core cells the inequality $u_i + v_j \le c_{ij}$ is satisfied for all $i = \overline{1, m}$, $j = \overline{1, n}$, then such reference plan is optimal.

The reference plan potentials are determined from the system equations $u_i + v_j = c_{ij}$ that write down for each filled cell of the transport task table.

Using the calculated potentials, we check the optimality condition $u_i + v_j \le c_{ij}$ for unfilled cells in the table.

If this condition is not met for at least one non-core cell, that is $u_i + v_j > c_{ij}$, the current plan is not optimal and you need to move to a new reference plan.

2. Transport task recalculation cycles

The transition from one reference plan to another is performed by filling a non-core cell for which the optimality condition is violated.

If there are several outlined cells, then the one that has the most disturbance is selected for filling, that is $\max\left\{\Delta_{ij} = (u_i + v_j) - c_{ij}\right\}$.

A recalculation cycle is built for the selected empty cell.

The redistribution of products within the cycle is carried out according to the following rules:

a) to each vertex of the cycle is assigned a specific sign, with an unfilled cell a sign "+", and all others alternately signs "-" and "+";

b) enter into the empty cell the least of the numbers in cells with the sign "-".

At the same time, this number is added to the corresponding numbers in cells with the sign "+" and subtracted from the numbers in cells with the sign "-".

Thus, the free cell became full and the corresponding cell with the smallest number (at the top with "-") became blank.

As a result of this redistribution of products, we will receive a new reference plan for the transport problem, which we again test for optimality.

If the solution is optimal, then we calculate the minimum value of the objective function (the minimum cost of transporting the entire load), and if not optimal, then again build the recalculation cycle and move to the new reference plan using the cycle shift.

The process is repeated until the optimal solution to the transport problem is obtained.

The values of the base cells that did not participate in the recalculation cycle remain unchanged in the new table.

If at some point the needs coincide with the stocks, then the needs are set to true zero, and in the stocks - dummy (zero with a dot).

Then a complete system of equations for finding the potential values is immediately constructed.

In degenerate plan problems, the cycle shift often needs to be set to 0.

Then, when moving to the next table, the values of the base cells that participated in the conversion cycle did not change.

Only this "base zero" is transferred to a non-base cell, for which it essentially did a recalculation cycle, and the cell that contained this 0 becomes free.

3 Statement of the Problem

Without reducing the generality of the approach proposed in the paper, let us consider the model problem of minimizing the cost of drinking water delivery.

Delivery of drinking water to vending machines will always be necessary, most people use them several times a week.

Therefore, there may be problems with the transportation and filling of the machines in a timely manner and with minimal gasoline costs.

The volume of drinking water is determined by the presence of a certain number of liters of water from the storage.

When delivering water to fill an automatic machine, there are situations where one machine can fill multiple automatic machines because it contains more water than necessary.

Alternatively, it is necessary for one machine to transport water to machines that are closest to each other and contain a volume so that the machine can fill them with the least residue.

When carrying out such operations with a large number of consumers, there are certain difficulties in finding the best option among the many existing ones that would provide for the needs of consumers and the costs of carriers would be minimal.

To solve this problem, you can build an economic-mathematical model that boils down to a transport-type problem.

Suppose we have n drinking water storage facilities that are in demand at a particular time, and it is planned to distribute water from them to m consumers (drinking water dispensers).

To construct an economic and mathematical model of the problem we introduce the following notation: i - water storage index, $i = \overline{1,n}$, j - vending machine index, $j = \overline{1,m}$, b_j - volume of water to be delivered to the j vending machine; c_{ij} - the cost of transporting water to the j vending machine from i storage; a_i - volume of water in the i storage in liters; x_{ij} - the number of liters that will be delivered to the vending machine from its storage.

It is necessary to find a solution to problem (1) - (3), that is, an optimal strategy for the distribution of available water resources, which will minimize the cost of the carrier during the water delivery.

4 Methods of Solving

In the paper, the potential method is chosen to parallel the solution of a given problem, because it allows to solve the transport problem for a finite number of steps (iterations) according to the constructed reference plan, which uses the least cost method [11-13].

The advantages of the potential method over other methods are that there is no need to construct cycles for each of the empty cells and simplify the calculation of the algebraic sums of tariffs.

Only one cycle is being built - the one that is being recalculated.

Applying the method of potentials, we can speak not of the sign of algebraic sums of tariffs, but of the comparison of indirect tariffs with the true ones.

The requirement for the inalienability of algebraic sums of tariffs is replaced by the condition that indirect tariffs do not exceed the true ones.

It should be borne in mind that the potentials (as well as cycles) for each new baseline plan are redefined.

To understand this, we first present a sequential algorithm for solving a problem, and then build a parallel algorithm based on it.

The sequential algorithm for solving the transport problem (1) - (3) by the potential method for obtaining the optimal solution is as follows:

1. Let $u_1 = 0, D_{\min} = c_{11} - u_1 - v_1$
2. Let's find out $u_i, v_j, i = \overline{2,n}, j = \overline{1,m}$.
3. For each cell (i, j) we calculate $D_{ij} = c_{ij} - u_i - v_j$, incidentally searching $D_{\min}$.
4. If not $D_{\min} \geq 0$, then the plan is optimal and the method ends.
5. We select the cell $(i_{\min}, j_{\min})$ with the smallest $D_{ij} = D_{\min}$ and construct the loop, while at the same time finding the least $x_{\min}$ of the values x_{ij} in the cells that have an even number in the cycle.

6. We assume that $x_{ij} = x_{ij} + x_{\min}$ if the sequence number of a cell in a loop is odd $x_{ij} = x_{ij} - x_{\min}$, if the sequence number of a cell in a cycle is even, then i, j - the coordinates of the cell.

7. Go to step 3.

Analyzing the algorithm for the possibility of parallelization, we came to the following conclusions:

1) Steps 1-2 are inappropriate to parallelize, since the potential value can only be obtained if the potential value is available and vice versa.

2) Steps 3-4 are convenient to parallelize since the shared data are arrays C, U and V, and do not change during execution, but are read-only.

The value $D_{\min}$ changes during execution, however, announcing local variables $D_{\min}$ for each of the threads allows for proper synchronization.

Therefore, steps 3-4 are subject to parallelization by distributing iterations between threads.

3) Step 5 is inappropriate to parallelize, the fragments of the search of the -this cell can only be performed if the -this cell of the cycle is present.

4) Step 6 is easy to parallelize using read-only shared data $x_{\min}$.

Therefore, step 6 is subject to parallelization by distributing iterations between threads.

So, the parallel algorithm will look like this:

Items 1), 2), 4), 6), 7), 9), 10) coincide with the sequential algorithm, and the following items are different:

3. We divide many cells into portions in proportion to the number of processor cores. We pass information to computing nodes.

5. Based on the data obtained from the computing nodes, we select the smallest $D_{\min}$.

8. We divide many cells into portions in proportion to the number of processor cores.

We pass the position of the cycle cells to the computing nodes.

5 Software Development

The program is designed using C ++.

C ++ compiled applications are mobile, namely it can be run on computers from different manufacturers and operating systems, making C ++ particularly popular.

Application interface (see Fig. 1)

1) Was designed in C ++ Builder6.

This allows you to edit the number of rows and columns of the matrix - add or delete them, fill in the data manually through the application interface, or read from a file.

In the absence of redundancy, no intermediate results are output, only a post-degeneration reference plan, U and V potentials, a free cell estimation matrix, an optimal plan and minimal transportation costs.

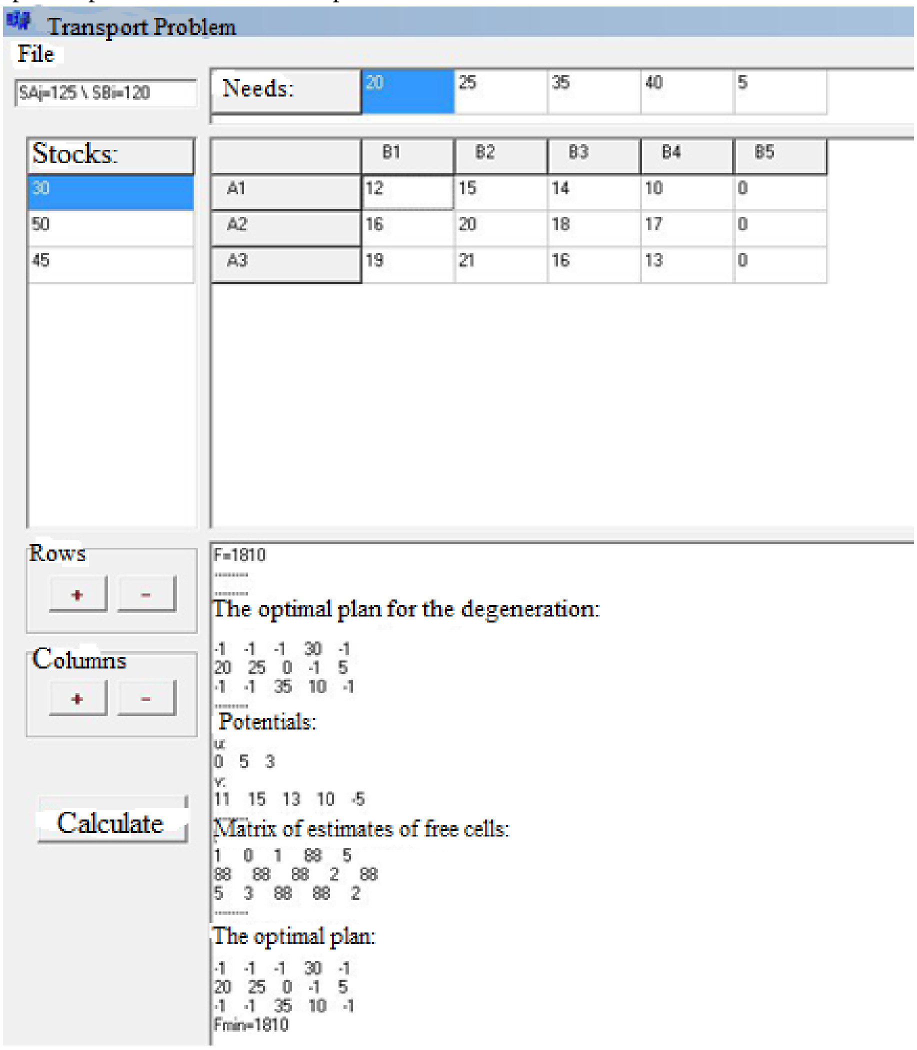
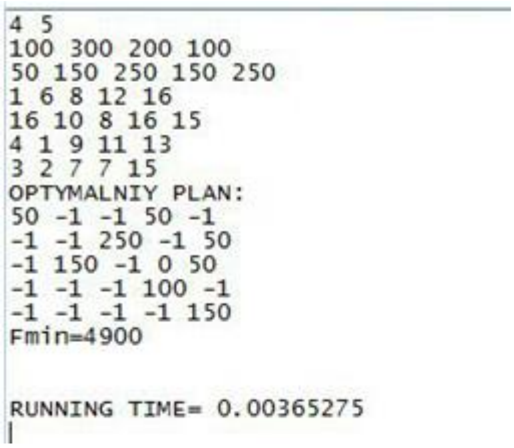


Fig. 1. Program interface

Also, we output all the intermediate and final results in a text file, in it you can see the dimension of the matrix, the initial matrix, all stages of its modification and such end results as the execution time, the reference plan and the minimum costs.

In Fig.2 you can see the initial matrix and the final results of the execution.



```
4 5
100 300 200 100
50 150 250 150 250
1 6 8 12 16
16 10 8 16 15
4 1 9 11 13
3 2 7 7 15
OPTYMALNIY PLAN:
50 -1 -1 50 -1
-1 -1 250 -1 50
-1 150 -1 0 50
-1 -1 -1 100 -1
-1 -1 -1 -1 150
Fmin=4900


RUNNING TIME= 0.00365275
```

Fig. 2. The file with the results of the program

To parallel the computational process, a parallel programming software with OpenMP specification was used [14].

OpenMP is a standard that includes compiler directives, libraries, and system variables that can be used to indicate concurrency on shared memory systems.

This is a technology that can be considered as a high-level add-on over Pthreads (or similar thread libraries).

OpenMP provides SPMD - a Model (Single Program Multiple Data) of parallel programming, which uses the same code for all parallel streams [14].

OpenMP is easy to use and includes only two basic types of designs: directives and functions of the OpenMP runtime that connect as an additional "omp.h" library. Graphic description language was used to represent the diagrams for object modeling in software development area - UML, namely in Draw IO.

6 Numerical Experiments

It is known [15] that the parallel acceleration is called the ratio of the execution time of a sequential program to the execution time of its parallel implementation:

$$R = \frac{t_{\text{consistent}}}{t_{\text{parallel}}} \tag{4}$$

After experimental verification of the execution time of the serial and parallel algorithm on the same input data, the average results of execution time and acceleration on several kernels are presented in Table 1.

From here you can see that the parallel algorithm gives an acceleration of more than one unit (gain) when the input matrix size is about 10,000 elements.

The graphs shown in Fig. 3 and Fig.4 it can be seen that, indeed, the larger the dimension, the better the advantage of the larger the number of processor cores.

Amdal's law [16] theoretically expects to obtain acceleration for a dual-core processor of 1,29 and for a quad-core one of 1,45.

Table 1. The average results of execution time and acceleration on several kernels

The dimension of the task, $m*n$	The time of execution of the sequential algorithm, s	Parallel algorithm execution time, s			
		2 processors		4 processors	
		Time	Acceleration	Time	Acceleration
100	0,0312	0,221	0,14	0,282	0,11
500	0,495	1,184	0,413	1,029	0,467
1000	1,003	1,562	0,642	1,483	0,676
10000	4,537	3,917	1,158	3,437	1,32
100000	14,527	11,91	1,219	10,15	1,431

Testing on the dimension of Problem 100000 obtained acceleration rates of 1,219 (2 processors) and 1,431 (4 processors).

This relatively low acceleration is related to the amount of concurrent code in the algorithm.

The results obtained confirm Amdal's law; the acceleration of the system cannot exceed the inverse of the proportion of successive calculations.

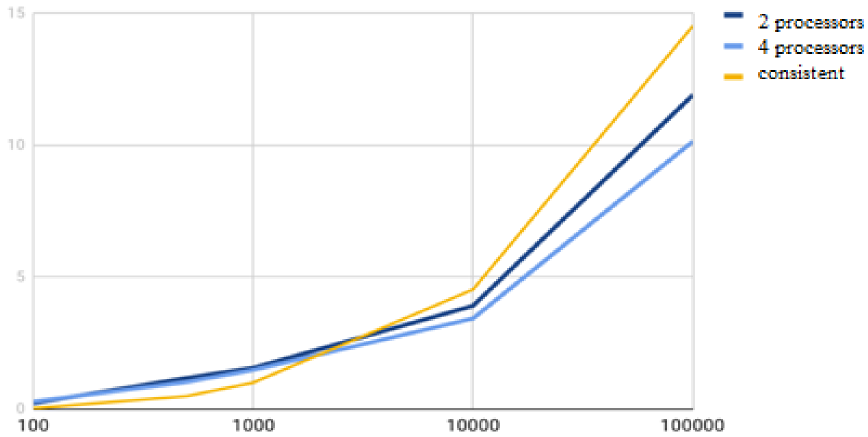


Fig. 3. Comparison of execution time of parallel and sequential algorithms

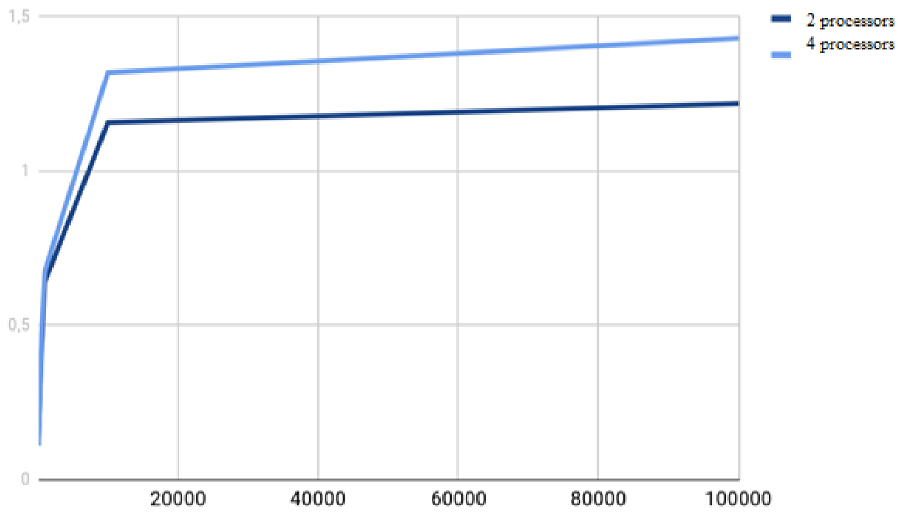


Fig. 4. Comparison of acceleration of serial and parallel algorithms

Conducted testing of the developed software allows to conclude that the parallel algorithm for solving the transport problem can be used to effectively find the optimal solution by the potential method only with a large amount of input data with an increased number of processor cores.

The latter is especially relevant for the modern development of multi-core systems.

7 Conclusion

In this work we propose an algorithm for finding the optimal solution of a transport problem that can be used on multi-core systems.

The latter are increasingly used in various subject areas and are a better opportunity to increase the power of processors.

The parallelization was done using OpenMP parallel programming technology.

The efficiency of this algorithm is tested by running it on dual- and quad-core processors.

The validity of the results obtained in the article is confirmed by the theoretically obtained estimates by Amdal's law.

You can also conclude that this algorithm allows you to get the best solution for a transport problem for large dimensions of input with greater efficiency.

This study is particularly relevant in the current rapid development of multi-core systems.

References

1. Ivanytska, O.V, Roshchina, N.V, Serbu,l R.S: Transport problem of linear programming, Agrosvit, No. 14, pp. 35-40 (2015)
2. Samoilenko, M.I, Skokov, B.G.: Operations Research (Mathematical Programming. Queuing Theory), 176 p., Kharkiv (2005)
3. Dunaevskaya, O.I.: Essence of mathematical methods and models for solving economic problems, International conferences: Research and optimization of economic processes "Optimum", pp. 128-134, Kharkov (2014)
4. Yakovlev, M.F., Gerasymova, T.O., Nesterenko, A.N.: Characteristic feature of the solving both of non-linear systems and systems of ordinary differential equations on parallel computer. In Proceedings of international symposium “Optimization problems of computations” (OPC - XXXV). Kyiv: V.M. Glushkov Institute of cybernetics of NAS of Ukraine, Vol. 2, pp. 435-439, Kyiv (2009).
5. Yakovlev, MV.F., Nesterenko, A.N., Brusnikin, V.N. Problems of the efficient solving of non-linear systems on multi-processor MIMD-architecture computers. Mathematical machines and systems. (4). P. 12-17. (2014).
6. Khymych, A.N., Molchanov, Y.N., Popov, A.V. other: Parallel algorithms for solving problems of computational mathematics. Kiev: Scientific Opinion, 248 pp. (2008).
7. Mochurad, L.I. Method of reduction model for calculation of electrostatic fields of electronic fields of electronic optics systems. Science journal Radioelektronika, informatics, management, 1(48), pp. 29-40 (2019). (Ukrainian)
8. Voss, M.: OpenMP Share Memory Parallel programming. Toronto, Kanada (2003).
9. Chapman, B., Jost, G.: Using OpenMP: portable shared memory parallel programming (Scientific and Engineering Computation). Cambridge, Massachusetts: The MIT Press (2008).
10. Chandra, R., Menon, R., Dagum, L., Kohr, D., Maydan, D., McDonald, J.: Parallel Programming in OpenMP. Morgan Kaufinann Publishers (2000).
11. Grama, A., Gupta, A., Karypis, G., Kumar, V.: Introduction to Parallel Computing. Addison Wesley, ISBN- 0-201-64865-2, 856 p. (2003).
12. Mochurad, L., Boyko, N.: Solving Systems of Nonlinear Equations on Multi-core Processors. Advances in Intelligent Systems and Computing IV Selected Papers from the International Conference on Computer Science and Information Technologies, CSIT 2019, September 17–20, pp. 90-106, Lviv, Ukraine (2019)
13. Sira, O.V, Bachkir, L.V.: Transport Problems with Stochastic Demand, KSPU Bulletin. Vol. 6/2006 (41). Part 1, pp. 22-25, Kyiv (2006)
14. Shakhovska, N., Boyko, N., Zasoba, Y., Benova, E.: Big data processing technologies in distributed information systems. Procedia Computer Science, 10th International conference on emerging ubiquitous systems and pervasive networks (EUSPN-2019), 9th International conference on current and future trends of information and communication technologies in healthcare (ICTH-2019), Vol. 160, 2019, pp. 561–566, Lviv, Ukraine (2019)
15. Boyko, N., Shakhovska, Kh., Mochurad, L., Campos, J.: Information System of Catering Selection by Using Clustering Analysis, Proceedings of the 1st International Workshop on Digital Content & Smart Multimedia (DCSMart 2019), pp. 94-106, Lviv, Ukraine (2019)
16. Kvurt, L., Tsylylyk, L.: The use of the laws of Amdal’s and Gustafson’s in estimating the acceleration rate in multiprocessor systems, Measurement Engineering and Metrology, Vol. 70, pp. 55-57, Lviv, Ukraine (2009)