

## ПАРАЛЕЛЬНІ АЛГОРИТМИ РОЗВ'ЯЗУВАННЯ СИСТЕМ НЕЛІНІЙНИХ РІВНЯНЬ ТА ЗАДАЧ КОШІ ДЛЯ СИСТЕМ ЗВИЧАЙНИХ ДИФЕРЕНЦІАЛЬНИХ РІВНЯНЬ НА БАГАТОЯДЕРНИХ КОМП'ЮТЕРАХ З ПРОЦЕСОРАМИ Intel Xeon Phi

*Т.О. Герасимова, А.Н. Нестеренко*

В роботі розглядаються алгоритми методів розв'язання систем нелінійних рівнянь (СНР) і задач Коші для систем звичайних диференціальних рівнянь (СЗДР) для багатоядерних комп'ютерів з процесорами Intel Xeon Phi. Наведено часи розв'язування СНУ і СЗДР різних порядків, обраховані коефіцієнти прискорення і ефективності використання запропонованих методів.  
Ключові слова: багатоядерні комп'ютери, системи нелінійних рівнянь, задачі Коші для систем звичайних диференціальних рівнянь.

В работе рассматриваются алгоритмы методов решения систем нелинейных уравнений (СНУ) и задач Коши для систем обыкновенных дифференциальных уравнений (СОДУ) для многоядерных компьютеров с процессорами Intel Xeon Phi. Приведены времена решения СНУ и СОДУ разных порядков, вычислены коэффициенты ускорения и эффективности использования предложенных методов.

Ключевые слова: многоядерные компьютеры, системы нелинейных уравнений, задачи Коши для систем обыкновенных дифференциальных уравнений.

The paper deals with algorithms of methods for the solving both of non-linear systems (NLS) and initial-value problems for systems of ordinary differential equations (SODE) on multi-core computers. Times required for the solving of various order SNE and SODE are given; acceleration and performance coefficients characterizing the employment of methods being proposed are evaluated, as well.

Key words: multi-core computers, non-linear systems, initial-value problems for systems of ordinary differential equations.

### Вступ

Нагальною проблемою сучасної науки є дослідження явищ та процесів різної природи. При цьому досить часто дослідження проводяться за допомогою чисельних експериментів на математичних моделях. Багато розрахункових задач, які виникають при математичному моделюванні, зводяться до розв'язання систем нелінійних та диференціальних рівнянь високого порядку. При розв'язуванні деяких задач виникає необхідність розв'язувати СНР або СЗДР швидше, ніж відбувається процес у реальному часі. Більш того, їх розв'язування потребує багатоваріантних розрахунків та значних обчислювальних ресурсів. Досить часто виникає необхідність розв'язувати задачі, для яких вихідні дані задані наближено. Тому для розв'язування таких задач та оцінки достовірності розв'язків доцільно використовувати високопродуктивні комп'ютери з паралельною організацією обчислень, в тому числі багатоядерні комп'ютери, та відповідні алгоритмічно-програмні засоби, які враховують архітектурні та обчислювальні особливості цих комп'ютерів.

### Постановка задач розв'язування СНР та задач Коші для СЗДР

**Задача для системи нелінійних рівнянь.** Нехай дана система  $n$  нелінійних рівнянь

$$f(x) = 0,$$

де  $x = (x_1, x_2, \dots, x_n)^T$ ,  $f(x) = (f_1(x), f_2(x), \dots, f_n(x))^T$  –  $n$ -вимірний вектор шуканого розв'язку та  $n$ -вимірна вектор-функція відповідно.

Задача (1) є деяким наближенням до точної системи нелінійних рівнянь  $\varphi(x)=0$ , і для цих вектор-функцій виконується нерівність:

$$\|f(u) - \varphi(u)\| \leq \delta$$

для будь-якого  $n$ -вимірного вектора  $u$ .

Для розв'язування задачі (1) задаються початкове наближення  $x^{(0)}$ , необхідна точність  $\varepsilon$  отримання наближення до розв'язку системи та визначається область, в якій шукається розв'язок,

$D = \{a_i \leq x_i \leq b_i, \quad i = 1, 2, \dots, n\}$ . При цьому початкове наближення належить визначеній області  $x^{(0)} \in D$ .

Нижнім індексом у формулах позначаються номери компонент векторів, а верхнім індексом будуть

позначатися номери ітерацій.

**Задача Коші для системи звичайних диференціальних рівнянь.** Нехай дано систему  $n$  звичайних диференціальних рівнянь. Задачу з початковими умовами (задачу Коші) для СЗДР  $n$ -го порядку на інтервалі  $[t_0, T]$  розглядатимемо у вигляді

$$\frac{dv}{dt} = \varphi(t, v),$$

$$v(t_0) = v^{(0)},$$

де  $v = (v_1, v_2, \dots, v_n)^T$  – шуканий вектор, а права частина системи –  $n$ -вимірний неперервний вектор-функція

$$\varphi(t, v) = (\varphi_1(t, v), \varphi_2(t, v), \dots, \varphi_n(t, v))^T.$$

При моделюванні реальних процесів на комп'ютері за допомогою СЗДР виникає ряд труднощів, зокрема, потреба мати справу з задачами з наближеними даними. Тому на практиці, як правило, замість задач (3), (4) маємо задачу з наближеними даними:

$$\frac{dy}{dt} = f(t, y),$$

$$y(t_0) = y^{(0)},$$

де  $y = (y_1, y_2, \dots, y_n)^T$ ,  $\|y^{(0)} - v^{(0)}\| \leq \delta$ ,  $f(t, y) = (f_1(t, y), f_2(t, y), \dots, f_n(t, y))^T$ , причому  $\varphi(t, w) = f(t, w) + \Delta(t, w)$ ,  $\|\Delta(t, w)\| \leq \Delta$  для довільної функції  $w(t)$ .

## Методи розв'язування СНР та задач Коші для СЗДР

**Метод Ньютона.** Якщо  $H = \left\{ \frac{\partial f_i}{\partial x_j} \right\}_{i,j=1}^n$  – матриця Якобі системи (1) (або деяке наближення до неї), то

ітераційний процес методу Ньютона знаходження розв'язку при заданому початковому наближенні записується у вигляді

$$H^{(k)} w^{(k)} = -f(x^{(k)}) \quad (7)$$

де  $w^{(k)} = x^{(k+1)} - x^{(k)}$  – поправка,  $k = 0, 1, \dots$  – номер ітерації, а  $x^{(k+1)} = x^{(k)} + w^{(k)}$ . Як видно з формули (7), на кожній ітерації необхідно обчислювати значення вектор-функції і матрицю Якобі та розв'язувати систему лінійних алгебраїчних рівнянь (СЛАР).

**Метод Рунге – Кутта 4-го порядку точності.** Одним із широко вживаних методів, що застосовується для чисельного інтегрування задач з початковими умовами для СЗДР, є метод Рунге – Кутта 4-го порядку.

Класичний метод Рунге – Кутта 4-го порядку реалізується за формулами:

$$y^{(i+1)} = y^{(i)} + (k_1 + 2k_2 + 2k_3 + k_4) / 6,$$

де ( $i = 0, 1, 2, \dots$ )

$$k_1 = h_i f(t_i, y^{(i)}),$$

$$k_2 = h_i f(t_i + h_i / 2, y^{(i)} + 0,5k_1),$$

$$k_3 = h_i f(t_i + h_i / 2, y^{(i)} + 0,5k_2),$$

$$k_4 = h_i f(t_i + h_i, y^{(i)} + 0,5k_3). \quad (9)$$

На кожному кроці інтегрування цей метод вимагає чотириразового обчислення вектор-функції  $f(t, y)$ , а для оцінки головного члена похибки, який визначає вибір кроку інтегрування, доводиться, частіше за все, тричі застосовувати метод Рунге – Кутта.

## Паралельні алгоритми розв'язування СНР та задач Коші для СЗДР

Розпаралелювання алгоритмів проводилось, виходячи з архітектури комп'ютера з багатоядерними процесорами Intel Xeon Phi. Кожний процесор Intel Xeon Phi має від 64 до 72 ядер, кожне з яких має два VPU (векторних процесорних пристроїв). Така архітектура процесора дає можливість мати декілька рівнів паралелізму.

Вищий рівень паралелізму передбачає використання середовища MPI, що дає можливість проводити обчислення у вигляді паралельних процесів на  $p$  ядрах, забезпечує синхронізацію обчислень та обмін інформацією між ядрами. На нижчому рівні кожний з таких процесів розпаралелюється між деякою кількістю потоків на вільних ядрах. Для розпаралелювання на цьому рівні доцільно використовувати програмні модулі математичної бібліотеки Intel MKL, які застосовуються для обчислення матрично-векторних операцій.

При розв'язуванні СНР та інтегруванні СЗДР будь-яким методом значна частина арифметичних операцій припадає на обчислення значень вектор-функції  $f(x)$ . Тому на комп'ютерах з паралельною організацією обчислень необхідно в першу чергу проводити розпаралелювання обчислення значень вектор-функції, яке дозволить автоматично розпаралелити обчислення наближення до матриці Якобі і розв'язування відповідної СЛАР та інтегрування СЗДР методом Рунге – Кутта за формулами (8), (9).

Автоматичний розподіл обчислення значень компонент деякої  $n$ -вимірної вектор-функції на  $p$  блоків ( $p$  – кількість процесів верхнього рівня паралелізму, що використовуються) виконується, виходячи з рівномірного завантаження процесів [1, 2].

**Паралельний алгоритм методу Ньютона.** На кожній ітерації алгоритму методу Ньютона виконуються наступні макрооперації:

- 1) обчислення кожним MPI-процесом частини компонент вектор-функції, що відповідають його логічному номеру;
- 2) обчислення кожним MPI-процесом відповідної кількості рядків матриці Якобі  $H(x^{(k+1)})$ ;
- 3) розв'язування отриманої СЛАР (7), використовуючи відповідний (до структури матриці Якобі) паралельний алгоритм, наприклад, паралельний алгоритм методу Гауса [3];
- 4) використовуючи отриманий розв'язок СЛАР, обчислення кожним MPI-процесом відповідних його логічному номеру компонент наступного наближення до розв'язку СНР  $x^{(k+1)} = x^{(k)} + w^{(k)}$ ;
- 5) перевірка умов закінчення ітераційного процесу за формулою

$$\|f(x^{(k+1)})\| \leq \frac{\varepsilon}{\|(H^{(k+1)})^{-1}\|},$$

яка є оцінкою якості наближеного розв'язку, що забезпечує виконання нерівності  $\|x^{(k+1)} - x\| \leq \varepsilon$ , де  $x$  – точний розв'язок СНР [4].

Значимо, що обчислення пунктів 2) – 5) проводиться з використанням функцій бібліотеки Intel MKL.

**Паралельний алгоритм методу Рунге – Кутта 4-го порядку.** Кожний  $i$ -й крок алгоритму реалізується за наступною схемою:

- 1) обчислення кожним MPI-процесом у відповідності з його логічним номером компонент векторів  $k_1, k_2, k_3, k_4$  та вектора  $y^{(i+1)} = y^{(i)} + v/6$ , де  $v = k_1 + 2k_2 + 2k_3 + k_4$ ;
- 2) за схемою першого пункту обчислення кожним MPI-процесом у відповідності з його логічним номером частину компонент  $\bar{y}^{(i+1)}$ , використовуючи двократне чисельне інтегрування СЗДР з кроком  $h_i/2$ ;
- 3) за схемою першого пункту обчислення кожним MPI-процесом у відповідності з його логічним номером частину компонент  $\bar{y}^{(i+1)}$ , використовуючи однократне чисельне інтегрування СЗДР з кроком  $h_i$ ;
- 4) обчислення кожним MPI-процесом у відповідності з його логічним номером частину компонент вектора похибки апроксимації системи  $\psi^{(i+1)} = \max_{1 \leq j \leq n} |\bar{y}_j^{(i+1)} - \bar{\bar{y}}_j^{(i+1)}|$ ;
- 5) обчислення кожним MPI-процесом при виконанні умов досягнення заданої точності уточненої довжини кроку інтегрування;

б) перевірка кожним MPI-процесом умов досягнення на наступному кроці інтегрування точки виводу розв'язку або кінцевої точки інтервалу інтегрування  $T$  та коригування при необхідності довжини кроку інтегрування.

Після обчислення розв'язку в точці виводу запам'ятовуються вектор розв'язку, константа Ліпшиця та оцінка похибки розв'язку [5].

Зазначимо, що обчислення пунктів 1) – 5) проводиться з використанням функцій бібліотеки Intel MKL.

### Експериментальне дослідження паралельних алгоритмів

Обчислювальні експерименти по розв'язуванню СНР та задач Коші для СЗДР проводилися на одноузловому (однопроцесорному) комп'ютері з процесором Intel Xeon Phi x200, який використовує 64 ядра.

**Експериментальне дослідження паралельного алгоритму методу Ньютона.** Методом Ньютона розв'язувалась система нелінійних рівнянь:

$$\sum_{j=1}^n x_j - 0.5(3n+1) + 2x_i^2 - 2 \left( 1 + 2 \frac{i}{n} + \left( \frac{i}{n} \right)^2 \right) = 0, \quad i=1, 2, \dots, n \quad (10)$$

при заданому початковому наближенні  $x^0 = 1 + \frac{i+1}{2n}$ , в області  $D = \{-1000 \leq x_i \leq 1000\}$ ,  $i = 0, 1, 2, \dots, n-1$ .

Наведемо деякі з отриманих результатів.

В табл. 1 представлені часи розв'язування СНР (10) в залежності від її порядку.

Таблиця 1

Кількість процесів	Час (сек.)		
	$n = 4\ 000$	$n = 5\ 000$	$n = 7\ 000$
1	806,92	1579,67	3652,44
8	53,76	214,23	497,31
16	39,71	111,39	162,67
32	32,73	68,71	154,23
64	30,93	52,21	136,44

Коефіцієнти прискорення  $S_p = T_1/T_p$ , де  $T_1$  та  $T_p$  часи розв'язування СНР відповідно на одному та  $p$  ядрах (рис. 1); та коефіцієнт ефективності  $E_p = S_p/p$  (рис. 2) для СНР (10) порядку  $n = 5\ 000$  показані на вказаних рисунках.

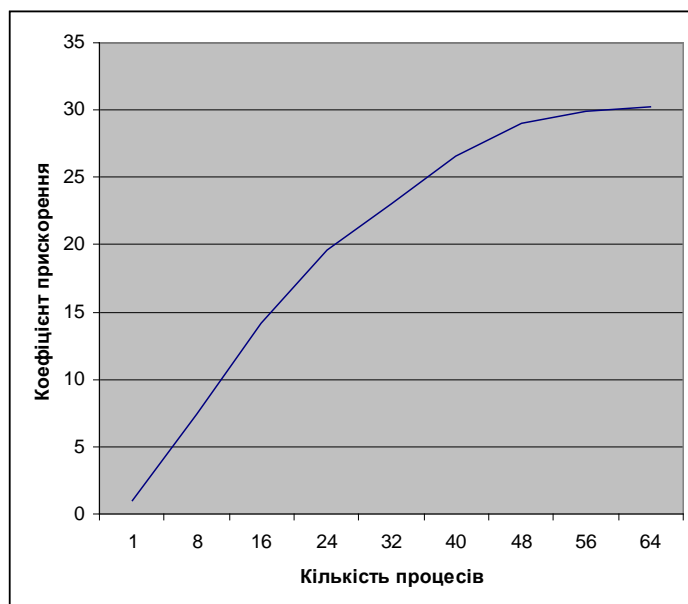


Рис. 1. Коефіцієнт прискорення

Результати, наведені на рис. 1, свідчать, що розроблений паралельний алгоритм забезпечує нормальну масштабованість, тобто час виконання задачі пропорційно зменшується з ростом кількості обчислювальних пристроїв. Найбільше прискорення досягається при використанні 64 процесів.

На наступному рисунку представлено значення коефіцієнту ефективності у відсотках, отриманого при розв'язуванні наведеної СНР п'ятитисячного порядку, з використанням паралельного алгоритму метода Ньютона.

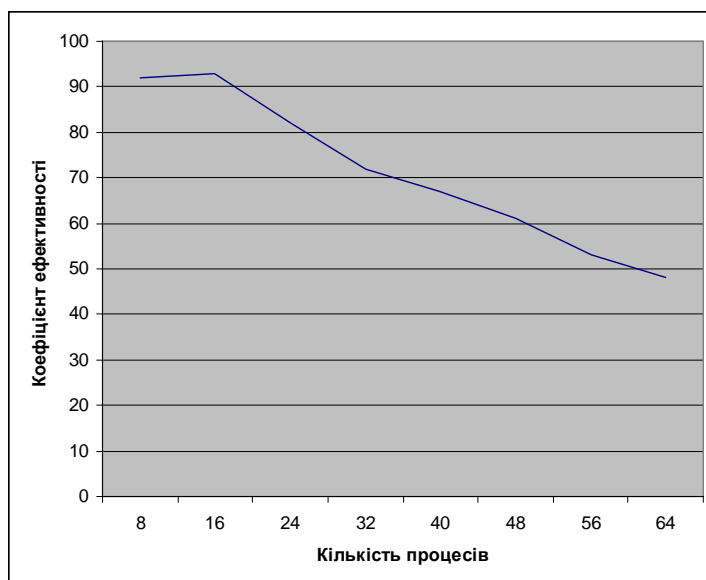


Рис. 2. Коефіцієнт ефективності

Наведені результати демонструють, що найбільша ефективність розв'язування системи п'ятитисячного порядку досягається при використанні 16 процесів.

**Експериментальне дослідження алгоритму методу Рунге – Кутта 4-го порядку.** Методом Рунге – Кутта 4-го порядку розв'язувалась задача Коші для системи звичайних диференціальних рівнянь:

$$\frac{du_i}{dt} = -\sum_{j=0}^{n-1} u_j - u_i + n(1+t) + 2+t$$

з початковими умовами  $u_i(0)=1 \quad i=0,1,2,\dots,n-1$ , на інтервалі  $[0,0; 0,4]$ .

Наведемо деякі з отриманих результатів.

В табл. 2 представлені часи розв'язування наведеної СЗДР в залежності від її порядку.

Таблиця 2

Кількість процесів	Час (сек.)		
	$n = 4\,096$	$n = 9\,600$	$n = 10\,752$
1	177,01	2 251,78	3 173,35
8	22,02	279,48	395,24
16	10,87	137,77	193,49
32	5,56	70,16	97,70
64	2,96	38,88	50,91

Коефіцієнти прискорення та ефективності для даної СЗДР порядку  $n = 9\,600$  показані на рис. 3 та рис. 4 відповідно.

Результати, показані на рис. 3, свідчать, що розроблений паралельний алгоритм забезпечує майже лінійне зростання коефіцієнта прискорення в залежності від кількості процесів. Найбільше прискорення досягається при використанні 64 процесів.

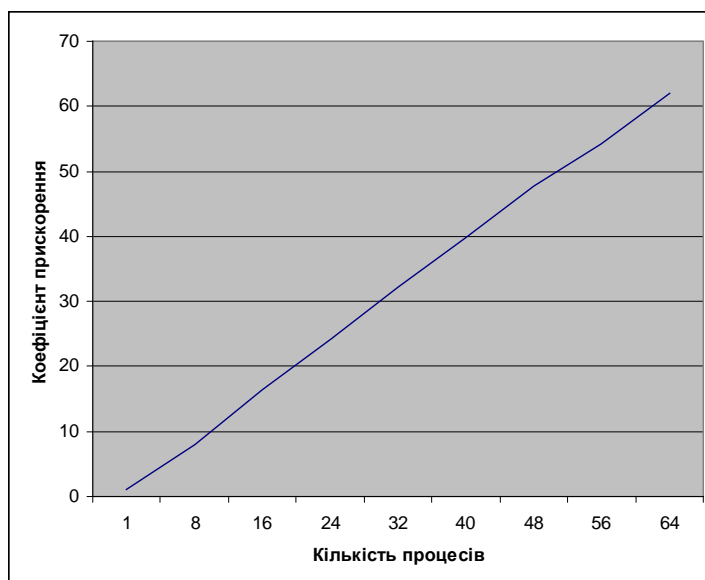


Рис. 3. Коефіцієнт прискорення

На рис. 4 показано значення коефіцієнту ефективності у відсотках, отриманого при розв'язуванні наведеної СЗДР з використанням паралельного алгоритму метода Рунге – Кутта 4-го порядку.

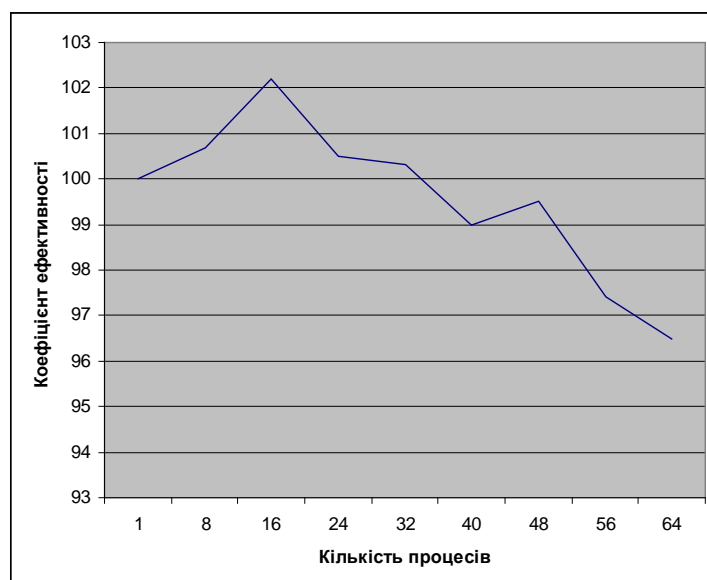


Рис. 4. Коефіцієнт ефективності

Отримані результати демонструють, що найбільша ефективність при розв'язуванні зазначеної СЗДР досягається при використанні 16 процесів. Отриманий коефіцієнт ефективності, який перевищує 100 %, свідчить, що паралельний алгоритм створено при правильному врахуванні особливостей архітектури комп'ютера.

## Висновки

Розроблені алгоритми розв'язування СНР та СЗДР великої розмірності призводять до значного скорочення часу їх розв'язання на комп'ютерах з процесорами Intel Xeon Phi. Отримані результати досягнуті за рахунок врахування архітектури процесора, яка дає можливість мати декілька рівнів паралелізму. Зауважимо, що розроблені програми призначено для однопроцесорного комп'ютера з процесорами Intel Xeon Phi, можна використовувати для багатопроцесорних комп'ютерів за умови розпаралелювання засобами MPI.

## Література

1. Яковлев М.Ф., Герасимова Т.О., Нестеренко А.Н. Особенности разв'язування систем нелинейных та дифференциальных уравнений на паралельных комп'ютерах. Питання оптимізації обчислень (ПОО – XXXV). Праці міжнародного симпозиуму. Київ: Інститут кібернетики імені В.М. Глушкова НАН України, 2009. Т. 2. С. 435–439.
2. Яковлев М.Ф., Нестеренко А.Н., Бруснікін В.М. Проблеми ефективного розв'язування систем нелинейных рівнянь на багатопроцесорних комп'ютерах MIMD-архітектури. Науково-теоретичний журнал "Математичні машини і системи" Київ. 2014. № 4. С. 12–17.
3. Химич А.Н., Молчанов И.Н., Попов А.В., Чистякова Т.В., Яковлев М.Ф. Параллельные алгоритмы решения задач вычислительной математики. К.: Наукова думка, 2008. 248 с.
4. Нестеренко А.Н., Химич А.Н., Яковлев М.Ф. Некоторые вопросы решения систем нелинейных уравнений на многопроцессорных вычислительных системах с распределенной памятью. Вестник компьютерных и информационных технологий. М.: 2006. № 10. С. 54–56.
5. Химич А.Н., Яковлев М.Ф., Герасимова Т.А. Некоторые вопросы решения систем обыкновенных дифференциальных уравнений на MIMD-компьютерах. Кибернетика и системный анализ. 2007. № 2. С. 175–182.

## References

1. Yakovlev, M.F. & Gerasymova, T.O. & Nesterenko, A.N. (2009) Characteristic features of the solving both of non-linear systems and systems of ordinary differential equations on parallel computers. In Proceedings of international symposium "Optimization problems of computations" (OPC – XXXV). Kyiv: V.M. Glushkov Institute of cybernetics of NAS of Ukraine, 2009. Kyiv: Vol. 2. P. 435–439.
2. Yakovlev, M.F. & Nesterenko, A.N. & Brusnikin, V.N. (2014) Problems of the efficient solving of non-linear systems on multi-processor MIMD-architecture computers. Mathematical machines and systems. (4). P. 12–17.
3. Khimich, A.N. et al. (2008) Parallel algorithms for the solving of computational mathematics problems. Kiev: Naukova Dumka.
4. Nesterenko, A.N. & Khimich, A.N. & Yakovlev, M.F. (2006) To the problem of solving of non-linear systems on multi-processor distributed memory computing system. Journal of computer and information technologies. 10. P. 54–56.
5. Khimich, A.N. & Yakovlev, M.F. & Gerasymova T.O. (2007) Some questions related to the solving of systems of ordinary differential equations on MIMD computers. Cybernetics and system analysis. (2). P. 175–182.

## Про авторів:

Герасимова Тетяна Олексіївна,

---

молодший науковий співробітник.  
Кількість наукових публікацій в українських виданнях – 30.  
<https://orcid.org/0000-0002-6319-5568>,

*Нестеренко Алла Никифорівна*,  
молодший науковий співробітник.  
Кількість наукових публікацій в українських виданнях – 30.  
<https://orcid.org/0000-0001-6174-1812>.

***Місце роботи авторів:***

Інститут кібернетики імені В.М. Глушкова НАН України,  
03187, Україна, Київ-187, проспект Академіка Глушкова, 40.  
Тел.: (066)-367-85-40.  
E-mail: [alla.nest1958@gmail.com](mailto:alla.nest1958@gmail.com).