УДК 004.272.2:519.63

О.А. Дмитрієва[1,2], д-р техн. наук, проф.,
Т.Г. Дараган[1], магістр
[1] Донецький національний технічний університет, м. Покровськ, Україна
[2] Дослідницький центр моделюючих технологій (SRC SimTech) університету Штуттгарта, Німеччина
olha.dmytriieva@donntu.edu.ua

# Розробка програмної системи паралельного моделювання динамічних об'єктів з оптимізацією комунікаційних операцій

*Роботу присвячено питанням розробки та тестування програмної системи, яка орієнтована на паралельне моделювання складних динамічних об'єктів зі сконцентрованими параметрами, що описуються задачею Коші у вигляді систем звичайних диференційних рівнянь (СЗДР) великої розмірності. Розробку програмної системи виконано згідно методології об'єктно-орієнтованого програмування з використанням сучасних програмних засобів. Проведено дослідження ефективності паралельних блокових методів з автоматичним керуванням кроку, запропоновано оптимізаційні схеми комунікаційних операцій, отримано оцінки локальних і глобальних похибок при розв'язанні тестових систем рівнянь, визначено вплив спроб збільшення або зменшення кроку інтегрування на всьому відрізку інтегрування на динаміку змін локальних похибок.*

***Ключові слова: задача Коші, паралельне моделювання, блокові методи, комунікаційні операції, керування кроком, MPI, похибка.***

*DOI: 10.31474/1996-1588-2017-2-25-69-75*

## *Вступ*

В даний час багато зусиль докладається для розробки і аналізу ефективних обчислювальних методів і алгоритмів, які спрямовані на дослідження динамічних об'єктів зі сконцентрованими параметрами, що описуються задачею Коші у вигляді систем звичайних диференційних рівнянь (СЗДР) великої розмірності. Реалізація великого класу таких задач, що характеризуються значною жорсткістю, поганю обумовленістю, розривністю і т.і. визначається неприйнятними часовими витратами. Цей факт обумовив необхідність залучання до розв'язання таких завдань обчислювальної техніки з паралельною або розподіленою архітектурою. [1-2]. Таким чином дана робота присвячена актуальним питанням ефективної структурної та алгоритмічної організації паралельних обчислень, розробці та обґрунтуванню методів вирішення широкого класу динамічних задач, які вимагають ефективного використання обчислювальних систем на багатопроцесорних ЕОМ, і є продовженням досліджень, результати яких відображено у [3-8]. В роботі в якості обчислювальних методів для моделювання поведінки динамічних систем використовуються блокові колокаційні методи [3-5], які характеризуються абсолютною стійкістю [3-4] та збіжністю за правою частиною і за початковими даними [3], що дозволяє використовувати їх при вирішенні жорстких задач, але разом з цим зростає складність побудови на їх основі алгоритмів автоматичної зміни розміру кроку інтегрування [4-5].

**Мета роботи** полягає у розробці та дослідженні ефективності програмної системи паралельного моделювання динамічних об'єктів з оптимізацією комунікаційних операцій для отримання максимальної реальної продуктивності.

**Задачею дослідження** є розробка паралельних різницевих схем з можливістю керування кроком і порядком інтегрування при вирішенні жорстких систем на основі паралельних колокаційних блокових методів і дослідження можливостей і особливостей відображення реструктуризованих алгоритмів на паралельні обчислювальні системи.

## *Управління кроком при чисельному інтегруванні блоковими методами*

В якості базового методу для управління кроком в даній роботі обирається схема збільшення опорного блоку, яку описано в [4-5], але на відміну від такого підходу, в роботі пропонується додаткові точки вводити в розрахунковий блок (рис. 1). Процедура управління кроком при збільшенням розрахункового блоку $(m \times (s+1))$ потребує декілька різницевих схем [7]. Це пояснюється тим, що для керування кроком, необхідно до початку основного розрахунку згенерувати різницеві схеми для постійного кроку, та додаткові схеми для інтегрування зі збільшеним та зменшеним кроком як для опорного так і для розрахункового блоків. Однак ці підготовчі заходи окупаються тим, що базові схеми розраховуються лише один раз для потрібного числа точок у першому та другому блоках, та використовуються на протязі усього процесу обчислення.

На рис. 1 зображено схему розрахунку $m$ – крокового $s$ точкового блокового методу, де блок розмірністю $(m \times s)$ обчислюється першою групою процесів, а блок розмірністю $(m \times (s+1))$ – другою групою процесів.

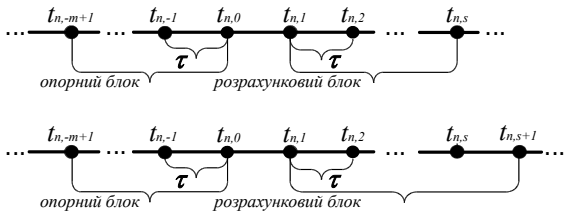


Рисунок 1 – Схема блочного методу зі збільшеним розрахунковим блоком

Оскільки обчислення проводяться для блоків точок, які розташовані регулярно, є можливість зіставлення розв'язків (рис. 2), отриманих методами с різними порядками точності в $s$ співпадаючих розрахункових точках $t_{n,1}, t_{n,2}, \dots, t_{n,s}$.

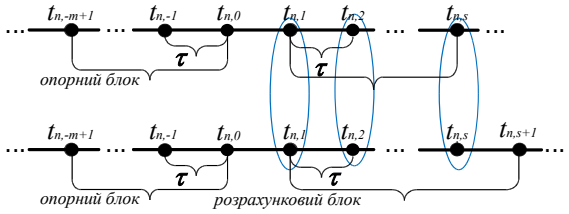


Рисунок 2 – Паралельні нитки обчислень m-крокових багатоточкових блокових методів розмірності $(m \times s)$ та $(m \times (s+1))$

На прийняття рішення щодо вибору розміру кроку впливає величина норми розбіжностей між значеннями наближених рішень в співпадаючих $s$ точках $t_{n,1}, t_{n,2}, \dots, t_{n,s}$ основного блоку з розмірністю $(m \times s)$, яка порівнюється із заданою величиною глобальної похибки $tol$, до якої додаються ще деякі запобіжні коефіцієнти, наприклад $\alpha_1 . \alpha_2$, такі, що $\alpha_1 \ll \alpha_2$. Якщо глобальна похибка з запобіжним коефіцієнтом $\alpha_1$ не перевищена, тобто

$$\vee u_{n,i}^{(m\times s)} - u_{n,i}^{(m\times(s+1))} \vee \leq \alpha_1 tol,$$

$i=1,2,..,s$, то наступний крок може бути збільшено або залишено без змін, в залежності від величини розбіжності, інформації про попередню варіацію кроку, та кількості точок, які включено у формування розв'язку ($s$ або $s$ +1). Визначається новий крок та новий вектор розв'язань для наступних блоків. Якщо

$$\alpha_1 tol < \vee u_{n,i}^{(m\times s)} - u_{n,i}^{(m\times(s+1))} \vee < \alpha_2 tol,$$

$i=1,2,..,s$,

то крок інтегрування залишається без змін і за основу береться розв'язання, отримане на блоці розмірністю $(m \times s)$. Зменшення кроку відбувається тоді, коли обчислена локальна похибка перевищує задану глобальну, тобто

$$\vee u_{n,i}^{(m\times s)} - u_{n,i}^{(m\times(s+1))} \vee \geq \alpha_2 tol, \; i=1,2,..,s,$$

тоді вектор розв`язків не формується, здійснюється перехід на базову схему зі зменшеним кроком. Саме такий підхід забезпечує розв'язання задач із заданою точністю необхідного порядку.

### *Оптимізація комунікаційних операцій при керуванні кроком*

Колокаційні блокові методи, які реалізовані за схемою збільшення точок у розрахункових блоках мають значні вимоги до реалізації їх на паралельних обчислювальних системах. У попередньому підрозділі описано алгоритми блочних методів розмірністю $(m \times s)$ та $(m \times (s+1))$, а також те, що кожен блок розраховується на окремій паралельній нитці, яка представлена групою процесів. Необхідність обміну даними між процесами виникає лише після закінчення розрахунку всіх точок в двох групах процесів (рис. 2). Звичайна реалізація такої схеми має на увазі, що кількість доступних процесорів розподіляється на дві групи. За кожною із груп закріпляється одна нитка обчислень, тобто один блок з різною кількістю точок.

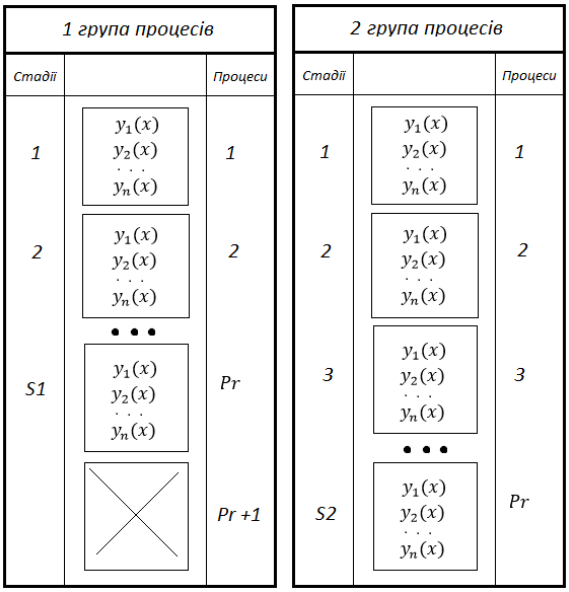


Рисунок 3 – Схема розподілу процесів на групи із виключенням надлишкових процесів

Такий підхід визначає, що за кожною точ-

кою в блоці закріпляється окремий процесор, який вираховує для однієї точки усю множину рівнянь зазначених у СЗДР. Після розрахунку усі процесори мають обмінятися своїми значеннями з усіма іншими, такий вид обміну також називають «усі з усіма». Такі обміни витрачають багато часу, та визначаються, як одна зі слабких ланок у подібних методах.

Однак можуть виникати такі ситуації, коли кількість доступних процесів значно більше, ніж число розрахункових точок в двох блоках, і тоді рівномірний розподіл процесорів на дві групи може бути неефективним. Для вирішення подібних ситуацій запропоновано два підходи для оптимального використання процесорів при вирішенні задач колокаційними блоковими методами. Перший підхід пропонується для використання на багатопроцесорних комп'ютерах, де кількість максимально доступних процесів не перевищує 20, саме такі показники є актуальними під час проведення досліджень. Суть підходу полягає у наступному: так як кількість процесів обмежена, то для того, щоб ефективно використати їх, необхідно задіяти наступний підхід для ефективного розподілення числа процесів на дві групи. Так як у головній першій групі число точок буде меншим, ніж у другому розрахунковому блоці, то кількість необхідних процесів для першої групи потрібно також скоротити до кількості точок у першому розрахунковому блоці. Тоді процеси, що залишилися за такою ж схемою розподіляються і для другої групи, і тоді незадіяні ПЕ просто не будуть приймати участі у розрахунку, кожен процес обчислює повністю всю систему ЗДР для своєї точки окремо, і час на обмін між процесами значно скоротиться. Такий підхід доцільно використовувати на архітектурі, яка описується топологією кільце.

На рис. 3 приведена схема розподілу процесів, де хрестиком позначено ті процеси, які не приймають участі в обчисленні.

Інший підхід розрахований на використання багатопроцесорних кластерів, де мінімальне число доступних процесів перевищує 100 процесів.

При використані таких обчислювальних ресурсів доцільніше буде задіяти усі процеси за наступним принципом. Клас задач, які обчислюються блочними методами характеризуються не тільки жорсткістю, а також великою розмірністю, яка може досягати і 1000, і 10000 рівнянь у системі. Тому обчислення усіх рівнянь на одному процесі, при загальній кількості більшої за 100, 200 не є ефективним, з точки зору часу виконання. Для подібних задач та кількості обчислювальних процесорів ефективним буде розподілити розрахунок усієї системи рівнянь на інші процесори. Розподіл процесів на дві групи виконується за такою схемою: усі доступні процеси рівномірно поділяються на загальне число точок у першому та другому розрахункових блоках, щоб за кожною точкою було закріплено підгрупу процесів, приблизно однакової розмірності. Саме такий підхід дозволить розпаралелювати обчислення всіх рівнянь системи на кожній стадії на декілька процесорів у кожній підгрупі. Таким чином при великій розмірності системи будуть задіянні всі доступні процеси.

На рис. 4 приведена схема розподілу процесів на дві групи для кожного блоку, а також розподіл на підгрупи для кожної обчислювальної точки у кожному блоці. Однак у зв'язку з тим, що на час проведення дослідження була відсутня можливість користуватися такими потужними обчислювальними системами, як кластер, на практиці було реалізовано перший підхід для оптимального використання процесорів при моделюванні динамічних об'єктів на основі колокаційних блокових методах з автоматичним управлінням кроку інтегрування, програмна реалізація та тестування яких описано в наступних підрозділах.

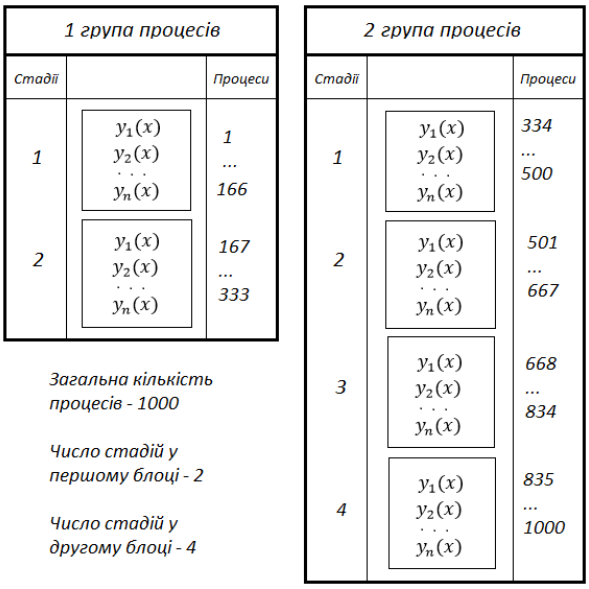


Рисунок 4 – Приклад розподілу великого числа процесів у кожному розрахунковому блоці

### *Опис структури програмної системи*

Для розробки програмної системи для паралельного обчислення СЗДР використовувалась сучасна популярна платформа Microsoft.NET та мова програмування C# у середовищі IDE MS Visual Studio 2013. Розпаралелювання даних було здійснено за допомогою бібліотеки MPI.NET, а відображення результатів обчислень на платформі .NET Framework у вигляді графіків здійснювалось з використанням окремого компоненту бібліотеки класів Windows Forms UserControl ZedGraph. Для розуміння поведінки та структури розроблювальної системи при проектуванні застосовувався сучасний засіб візуального моделювання з викорис-

танням нотації UML (Unified Modeling Language).

Розроблена програмна система поділяється на дві окремі підсистеми: розрахункову, та підсистему візуалізації, яка детально відображує результати розрахунків. Запустивши першу підпрограму, яка відповідає за обчислення, необхідно обрати кількість процесів, що будуть задіяні у роботі програми. Програма може буде запущена в розподіленій системі з середовищем MPI на необхідній кількості процесорів. Структура програмної системи містить наступні модулі:

– головний модуль, який відповідає за збір початкових даних від користувача та запуск обраної задачі на заданому числі процесорів;

– модуль функцій, який включає головні функції програмної системи, а саме: генерацію початкових різницевих коефіцієнтів, обчислення кроку, обмін повідомленнями між процесорами;

– модуль з послідовною реалізацією блокових методів для розв'язання систем диференційних рівнянь для тестових задач;

– модуль з паралельною реалізацією блокових методів.

Після закінчення розв'язання задачі дані результатів зберігаються у вихідний файл, тому користувач має можливість переглянути отримані дані, запустивши для цього другу підпрограму системи. Для перегляду результатів обчислення необхідно обрати файл з результатами та відкрити їх у програмі. Вихідний файл містить результати обчислень для детального аналізу: кількість задіяних процесорів, кількість відкинутих кроків, кількість прийнятих кроків, результати обчислень у вигляді одновимірного масиву.

Друга підсистема складається з одного модулю, головна функція якого полягає у зчитуванні результатів з файлу та відображенні їх в графічному вигляді. На рис. 5 зображена діаграма класів програмної системи, що спроектовано.

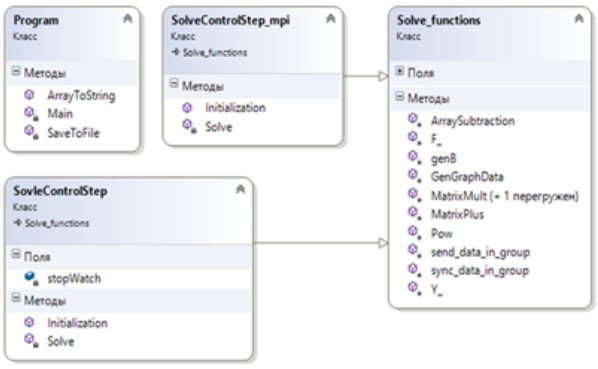


Рисунок 5 – Діаграма класів

Головний клас Program, який виконується першим при запуску підсистеми для обчислення, отримує початкові данні від користувача, передає ці данні до необхідного класу, в залежності від обраного методу розв'язання задачі, та запускає процес обчислення. Якщо було обрано послідовний алгоритм обчислення, то данні передаються до класу SolveControlStep, при обранні паралельного алгоритму обчислення данні передаються до класу SolveControlStep_mpi.

У класі Solve_function об'явлені атрибути (поля), які наслідуються класами SolveControlStep та SolveControlStep_mpi. Також клас Solve_function містить методи, які є загальними як для послідовного, так і для паралельного алгоритмів обчислення, лише з однією різницею, яка полягає у тому, що методи send_data_in_group() та sync_data_in_group() використовуються у паралельному алгоритмі для передачі повідомлень між задіяними процесами. Класи SolveControlStep та SolveControlStep_mpi відрізняються також тим, що для ініціалізації розрахунку аналогічні методи приймають різну кількість аргументів, тому що для паралельного обчислення необхідні додаткові початкові данні, які стосуються кількості процесорів.

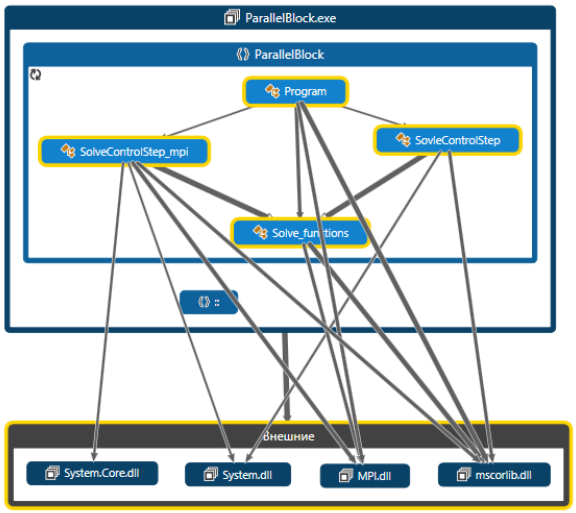


Рисунок 6 – Діаграма залежностей класів системи

Використання зовнішніх бібліотек в програмній системі, що проектується, має на увазі те, що існує пряма залежність між класами програмної системи та цими бібліотеками. Відображення таких залежностей на діаграмі допомагає визначитися з тим, як буде впливати зміна структури програмної системи на зв'язки з іншими компонентами, що, в свою чергу, є допоміжним засобом при модифікації цієї самої системи.

На рис. 6 зображена діаграма залежностей між класами у програмній системі, а також їх зв'язки з зовнішніми бібліотеками.

### *Аналіз ефективності моделювання для паралельних обчислень*

При проведені тестування блоковими методами обчислювалась задача Капса [9] виду

$$x_1'(t) = -(\lambda + 2)x_1(t) + \lambda x_2^2(t), x_1(0) = 1,$$

$$x_2'(t) = x_1(t) - x_2(t)\big(1 + x_2(t)\big), x_2(0) = 1,$$
$$0 \le t \le 10, (1)$$

з відомими точними розв'язаннями:

$$x_1(t) = e^{-2t}, x_2(t) = e^{-t}. (2)$$

До початку проведення обчислень було згенеровано розрахункові коефіцієнти для опорних та розрахункових блоків. Також для запуску процесу обчислень було застосовано варіативні значення коефіцієнту жорсткості λ, початкового кроку інтегрування $\tau_1$, коефіцієнту трудомісткості реалізації правих частин (коефіцієнту навантаження).

Тестування програмної системи з паралельним алгоритмом розв'язання задачі Коші для СЗДР виконувалося на комп'ютері з 4 доступними процесами, тому результати обчислень, а саме, час розрахунку тих задач, де число заданих процесів дорівнює 6 та 8, значно вище, ніж при обчисленні 2 або 4 процесами. Це пояснюється тим, що операційна система додатково розподіляє ті обчислення, які були закріплені за 5, 6 та іншими процесами, на доступні 4, використовуючи потоки. Однак отримані результати все одно дозволяють оцінити ефективність проведених обчислень.

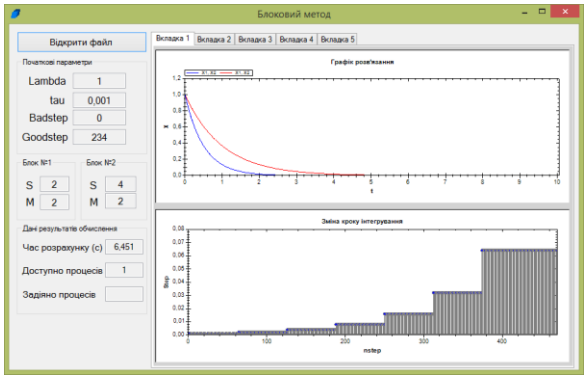


Рисунок 7 – Результати обчислення системи при λ=1

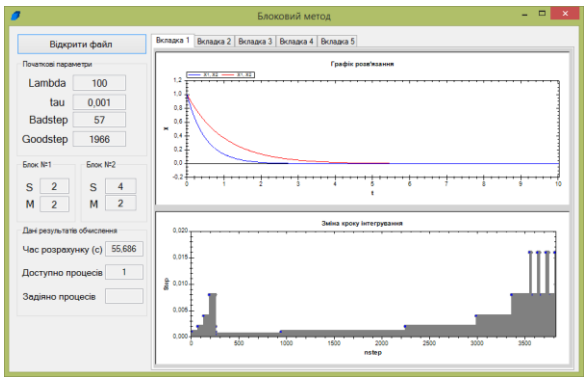


Рисунок 8 – Результати обчислення системи при λ=100

На рис. 7-8 відображено результати обчислень системи рівнянь (1) з коефіцієнтами жорсткості λ=1 та λ =100, та коефіцієнтом навантаження 5000. Порівнюючи отримані результати, видно, що при обчисленні системи рівнянь з λ=1 не було відкинутих кроків, а навпаки, крок інтегрування постійно зростав. Однак при λ=100 з'явились відкинуті кроки інтегрування, та були спроби їх зменшення, що означає збільшення загальної кількісті ітерацій при обчисленні кроків.

Графіки локальної похибки при обчисленні системи рівнянь на 4 процесах зображено на рис. 9, який відображає спроби збільшення або зменшення кроку інтегрування на всьому відрізку.

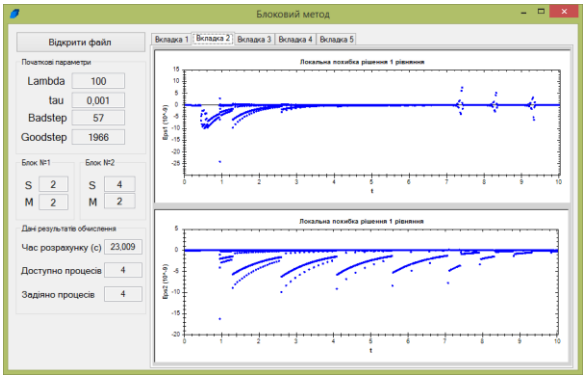


Рисунок 9 – Графіки локальної похибки для 1 та 2 рівнянь системи

## *Висновки*

В роботі в якості обчислювальних методів для моделювання поведінки динамічних систем використовувалися блокові колокаційні методи, які характеризуються абсолютною стійкістю та збіжністю за правою частиною і за початковими даними. Проведено дослідження ефективності паралельних блокових методів з автоматичним керуванням кроку. Розглянуто проблему вибору ефективної кількості обчислювальних вузлів для паралельного розв'язання блоковими методами. Розроблено оптимізаційні схеми комунікаційних операцій при керуванні кроком інтегрування. Проектування програмної системи здійснювалося на підставі засобів візуального моделювання з використанням діаграм UML. Реалізовано програмну систему, що включає реалізацію послідовних, паралельних та модифікованих паралельних алгоритмів розв'язання СЗДР блоковими методами, множину тестових завдань для проведення обчислювальних експериментів. Розробку програмної системи виконано згідно методології об'єктно-орієнтованого програмування з використанням сучасних програмних засобів та інструкцій MPI.

При проведені експериментів блоковими

методами використовувалася задача Капса з відомими точними розв'язаннями. Проаналізовано отримані результати експериментальних обчислень, які дозволили зробити висновок, що запропонований та досліджений метод оптимізації комунікаційних операцій керуванням кроку в паралельних алгоритмах моделювання мінімізує обчислювальні ресурси при вирішенні систем диференціальних рівнянь, та може бути використаний при подальшому дослідженні оптимізації паралельних алгоритмів розв'язання СЗДР на основі блокових методів.

**Наукова** новизна полягає у нових підходах до оптимізації комунікаційних операцій управління кроком в паралельних алгоритмах моделювання.

**Практичне значення** полягає в підвищенні ефективності використання паралельних обчислювальних систем при вирішенні динамічних завдань великої розмірності з зосередженими параметрами. Отримані алгоритми управління кроком і порядком інтегрування при вирішенні жорстких систем з генеруванням коефіцієнтів заданого порядку точності дозволили прискорити вирішення жорстких задач з контролем локальної похибки на паралельних обчислювальних системах MIMD (multiple instruction multiple data) типів з різними топологічними характеристиками і типами пам'яті.

### *Список літератури*

1. Migdalas A. Parallel Computing in Optimization / A. Migdalas, M. Panos Pardalos, S. Storoy. – Berlin-Heidelberg: Springer Verlag, 2012. – 608 p. – ISBN 978-1461334026

2. Kahaner D.K. Experiments with an ordinary differential equation solver in the parallel solution of method of lines problems on a sharedmemory parallel computer / D.K. Kahaner, E. Ng, W.E. Schiesser, S. Thompson// Journal of Computational and Applied Mathematics 38 (1991) 231-253

3. Дмитриева О.А. Параллельные численные методы моделирования динамических объектов: монография / О.А. Дмитриева. – Красноармейск: ГВУЗ «ДонНТУ», 2016. – 384 с

4. Дмитриева О.А. О построении параллельных разностных схем моделирования с вариацией шага в расчетном блоке / Дмитриева О.А // Вестник НТУ «ХПИ». Серия: Информатика и моделирование. – Харьков: НТУ «ХПИ». – 2016. – № 21(1193). – С. 20 – 28.

5. Дмитриева О.А. Коллокационные блочные методы с контролем на шаге / О. А. Дмитриева //Системи обробки інформації. – 2015, № 5(130). – С. 78-84.

6. Дмитрієва О.А. Дослідження ефективності програмної системи паралельного моделювання динамічних об'єктів / О. А. Дмитрієва, Т. Г Дараган // Матеріали 7-ої міжнародної науково-технічної конференції " Моделювання і комп'ютерна графіка ", 18 ─ 24 вересня 2017 р. (МКГ-2017), Покровськ. – 2017. – С. 49-55.

7. Дмитриева О. А. Моделирование жестких систем на основе коллокационных схем растяжения – сжатия / О. А. Дмитриева, Н. Г. Гуськова // Наукові праці Донецького національного технічного університету. Серія: Інформатика, кібернетика та обчислювальна техніка. - 2017. - № 1 (24). - С. 76-84.

8. Dmitrieva O. Parallel Step Control. Development of parallel algorithms of the step variation for simulation of stiff dynamic systems / O. Dmitrieva, L. Feldman. – Lambert Academic Publishing. – 2013. – 72p.

9. Kaps P. A study of Rosenbrock-type methods of high order / P.Kaps, G. Wanner // Numerische Mathematik, vol. 38, no. 2, pp. 279–298, 1981.

### *References*

1. Migdalas, A., Panos Pardalos, M., Storoy, S. (2012), "Parallel Computing in Optimization", Berlin-Heidelberg: Springer Verlag, 608 p.

2. Kahaner, D.K., Ng, E., Schiesser, W.E., Thompson, S. (1991), "Experiments with an ordinary differential equation solver in the parallel solution of method of lines problems on a sharedmemory parallel computer", Journal of Computational and Applied Mathematics, P. 231-253.

3. Dmitrieva, O. (2016), "Parallel numerical methods for modeling dynamic objects", [Parallel'nye chislennye metody modelirovanija dinamicheskih ob"ektov], Pokrovsk: «DonNTU», 384 p.

4. Dmitrieva, O. (2016), "On the construction of parallel difference modeling schemes with the variation of the step in the calculation block", [O postroenii parallel'nyh raznostnyh shem modelirovanija s variaciej shaga v raschetnom bloke], Har'kov: NTU «HPI», P. 20 – 28.

5. Dmitrieva, O. (2015), "Collocations block methods with step control", [Kollokacionnye blochnye metody s kontrolem na shage], Information processing systems, P. 78-84.

6. Dmitrieva, O., Feldman, L. (2013), "Parallel Step Control. Development of parallel algorithms of the step variation for simulation of stiff dynamic systems", Lambert Academic Publishing, 72p.

7. Dmitrieva, O., Daragan, T. (2017), "Investigation of the effectiveness of the software system for parallel modeling of dynamic objects", [Issledovanie jeffektivnosti programmnoj sistemy parallel'nogo modelirovanija dinamicheskih ob"ektov], Materials of the Seventh International Scientific and Technical Conference "Modeling and Computer Graphics", P. 49-55.

8. Dmitrieva, O., Huskova, N. (2017), "Simulation of rigid systems based on collocation stretching-compression schemes", [Issledovanie jeffektivnosti programmnoj sistemy parallel'nogo modelirovanija dinamicheskih ob"ektov], Scientific works of Donetsk National Technical University. Series: Computer Science, Cybernetics and Computer Science, P. 76-84.

9. Kaps, P., Wanner, G. (1981), "A study of Rosenbrock-type methods of high order", Numerische Mathematik, P. 279–298.

*Надійшла до редакції 15.12.2017*

**О.А. ДМИТРИЕВА[1,2], Т.Г. ДАРАГАН[1]**
[1]Донецкий национальный технический университет, г. Покровск, Украина
[2]Исследовательский центр моделирующих технологий (SRC SimTech) университета Штутгарта, Германия

**РАЗРАБОТКА ПРОГРАММНОЙ СИСТЕМЫ ПАРАЛЛЕЛЬНОГО МОДЕЛИРОВАНИЯ ДИНАМИЧЕСКИХ ОБЪЕКТОВ С ОПТИМИЗАЦИЕЙ КОММУНИКАЦИОННЫХ ОПЕРАЦИЙ**

Работа посвящена вопросам разработки и тестирования программной системы, ориентированной на параллельное моделирование сложных динамических объектов с сосредоточенными параметрами, описываемых задачей Коши в виде систем обыкновенных дифференциальных уравнений (СОДУ) большой размерности. Разработку программной системы выполнено согласно методологии объектно-ориентированного программирования с использованием современных программных средств. Проведено исследование эффективности параллельных блочных методов с автоматическим управлением шага, предложены оптимизационные схемы коммуникационных операций, получены оценки локальных и глобальных ошибок при решении тестовых систем уравнений, определено влияние попыток увеличения или уменьшения шага интегрирования на всем отрезке интегрирования на динамику изменений локальных погрешностей.

***Ключевые слова: задача Коши, параллельное моделирование, блочные методы, коммуникационные операции, управление шагом, MPI, погрешность***

**O. DMITRIEVA[1,2], T. DARAGAN[1]**
[1]Donetsk National Technical University, Pokrovsk, Ukraine
[2]Stuttgart Research Centre for Simulation Sciences (SRC SimTech), University Stuttgar

**DEVELOPMENT OF THE PROGRAM SYSTEM OF PARALLEL SIMULATION OF DYNAMIC FACILITIES WITH OPTIMIZATION OF COMMUNICATION OPERATIONS**

The work is devoted to the development and testing of a software system oriented to the parallel simulation of complex dynamic objects with lumped parameters, described by the Cauchy problem in the form of systems of ordinary differential equations (SODE) of large dimension. The development of the software system is performed according to the methodology of object-oriented programming using modern software. The efficiency of parallel block methods with automatic step control has been investigated, optimization schemes of communication operations have been proposed, estimates of local and global errors in solving test systems of equations have been obtained, and the influence of attempts to increase or decrease the integration step on the entire integration interval on the dynamics of changes in local errors has been determined.

In the work as computational methods for simulation of the behavior of dynamic systems, block collocation methods were used, which are characterized by absolute stability and convergence on the right side and on the initial data. The study of the efficiency of parallel block methods with automatic step control is carried out. The problem of choosing an effective number of computing nodes for parallel solution by block methods is considered. The optimization schemes of communication operations in the control of the integration step are developed. The design of the software system was carried out on the basis of visual simulation tools using UML diagrams. A software system has been implemented that includes the implementation of successive, parallel and modified SODE algorithms by block methods, as well as a number of test tasks for conducting computational experiments. The development of the software system is carried out in accordance with the methodology of object-oriented programming using modern software tools and MPI instructions.

When conducting experiments by block methods, the Kapsa problem with known exact solutions was used. The obtained results of experimental calculations were analyzed, which made it possible to conclude that the proposed and investigated method of optimizing communication operations in step management in parallel simulation algorithms minimizes computational resources in solving systems of differential equations and can be used in further researches on optimization of parallel algorithms for SODE based on block methods.

***Key words: Cauchy problem, parallel simulation, block methods, communication operations, step control, MPI, error.***